Estimation de posture 3D à partir de données imprécises et incomplètes : application à l'analyse d'activité d'opérateurs humains dans un centre de tri

Thibault Blanc-Beyne

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Institut National Polytechnique de Toulouse (Toulouse INP)

Discipline ou spécialité :

Informatique et Télécommunication

Présentée et soutenue par :

M. THIBAULT BLANC-BEYNE

le lundi 9 novembre 2020

Titre :

Estimation de posture 3D à partir de données imprécises et incomplètes : application à l'analyse d'activité d'opérateurs humains dans un centre de tri

---

Ecole doctorale :

Mathématiques, Informatique, Télécommunications de Toulouse (MITT)

Unité de recherche :

Institut de Recherche en Informatique de Toulouse (IRIT)

Directeur(s) de Thèse :

M. VINCENT CHARVILLAT

Rapporteurs :

M. FRANCK MULTON, UNIVERSITE RENNES 2

M. FRANÇOIS BREMOND, INRIA SOPHIA ANTIPOLIS

Membre(s) du jury :

Mme STÉFANIE HAHMANN, INP GRENOBLE, Président

M. ALAIN FLUHR, EBHYS SAS, Invité

M. AXEL CARLIER, TOULOUSE INP, Membre

Mme IRENE GAILLARD, CENTRE REGIONAL CNAM, Membre

Mme NATALIA NEVEROVA, FACEBOOK FRANCE, Membre

Mme SANDRINE MOUYSSET, UNIVERSITE TOULOUSE 3, Membre

M. PHILIPPE MILLES, EBHYS SAS, Invité

M. VINCENT CHARVILLAT, TOULOUSE INP, Membre

Titre : Estimation de posture 3D à partir de données imprécises et incomplètes : application à l'analyse d'activité d'opérateurs humains dans un centre de tri

Résumé :
Dans un contexte d'étude de la pénibilité et de l'ergonomie au travail pour la prévention des troubles musculo-squelettiques, la société Ebhys cherche à développer un outil d'analyse de l'activité des opérateurs humains dans un centre de tri, par l'évaluation d'indicateurs ergonomiques. Pour faire face à l'environnement non contrôlé du centre de tri et pour faciliter l'acceptabilité du dispositif, ces indicateurs sont mesurés à partir d'images de profondeur.

Une étude ergonomique nous permet de définir les indicateurs à mesurer. Ces indicateurs sont les zones d'évolution des mains de l'opérateur et d'angulations de certaines articulations du haut du corps. Ce sont donc des indicateurs obtenables à partir d'une analyse de la posture 3D de l'opérateur. Le dispositif de calcul des indicateurs sera donc composé de trois parties : une première partie sépare l'opérateur du reste de la scène pour faciliter l'estimation de posture 3D, une seconde partie calcule la posture 3D de l'opérateur, et la troisième utilise la posture 3D de l'opérateur pour calculer les indicateurs ergonomiques.

Tout d'abord, nous proposons un algorithme qui permet d'extraire l'opérateur du reste de l'image de profondeur. Pour ce faire, nous utilisons une première segmentation automatique basée sur la suppression du fond statique et la sélection d'un objet dynamique à l'aide de sa position et de sa taille. Cette première segmentation sert à entraîner un algorithme d'apprentissage qui améliore les résultats obtenus. Cet algorithme d'apprentissage est entraîné à l'aide des segmentations calculées précédemment, dont on sélectionne automatiquement les échantillons de meilleure qualité au cours de l'entraînement.

Ensuite, nous construisons un modèle de réseau de neurones pour l'estimation de la posture 3D de l'opérateur. Nous proposons une étude qui permet de trouver un modèle léger et optimal pour l'estimation de posture 3D sur des images de profondeur de synthèse, que nous générons numériquement.

Finalement, comme ce modèle n'est pas directement applicable sur les images de profondeur acquises dans les centres de tri, nous construisons un module qui permet de transformer les images de profondeur de synthèse en images de profondeur plus réalistes. Ces images de profondeur plus réalistes sont utilisées pour réentrainer l'algorithme d'estimation de posture 3D, pour finalement obtenir une estimation de posture 3D convaincante sur les images de profondeur acquises en conditions réelles, permettant ainsi de calculer les indicateurs ergonomiques.

Title : Estimation of 3D pose from inaccurate and incomplete data : application to human operators' activity analysis in a sorting center

Abstract :
In a context of study of stress and ergonomics at work for the prevention of musculoskeletal disorders, the company Ebhys wants to develop a tool for analyzing the activity of human operators in a waste sorting center, by measuring ergonomic indicators. To cope with the uncontrolled environment of the sorting center, these indicators are measured from depth images.

An ergonomic study allows us to define the indicators to be measured. These indicators are zones of movement of the operator's hands and zones of angulations of certain joints of the upper body. They are therefore indicators that can be obtained from an analysis of the operator's 3D pose. The software for calculating the indicators will thus be composed of three steps : a first part segments the operator from the rest of the scene to ease the 3D pose estimation, a second part estimates the operator's 3D pose, and the third part uses the operator's 3D pose to compute the ergonomic indicators.

First of all, we propose an algorithm that extracts the operator from the rest of the depth image. To do this, we use a first automatic segmentation based on static background removal and selection of a moving element given its position and size. This first segmentation allows us to train a neural network that improves the results. This neural network is trained using the segmentations obtained from the first automatic segmentation, from which the best quality samples are automatically selected during training.

Next, we build a neural network model to estimate the operator's 3D pose. We propose a study that allows us to find a light and optimal model for 3D pose estimation on synthetic depth images, which we generate numerically. However, if this network gives outstanding performances on synthetic depth images, it is not directly applicable to real depth images that we acquired in an industrial context.

To overcome this issue, we finally build a module that allows us to transform the synthetic depth images into more realistic depth images. This image-to-image translation model modifies the style of the depth image without changing its content, keeping the 3D pose of the operator from the synthetic source image unchanged on the translated realistic depth frames. These more realistic depth images are then used to re-train the 3D pose estimation neural network, to finally obtain a convincing 3D pose estimation on the depth images acquired in real conditions, to compute de ergonomic indicators.

Remerciements

Sans un certain nombre de personnes pour m'accompagner, je n'aurais pas pu réaliser les travaux présentés dans ce manuscrit. Je tiens donc à leur témoigner toute ma gratitude.

Pour commencer, je souhaite remercier Stefanie Hahmann d'avoir accepté d'être membre de mon jury de thèse et de le présider, mais aussi car c'est elle qui m'a conseillé de venir faire mon projet de fin d'études à Toulouse au sein de l'équipe dans laquelle j'ai ensuite réalisé ma thèse.

Ensuite, je voudrais remercier François Brémond et Franck Multon pour avoir accepté d'être rapporteurs ce manuscrit de thèse. J'ai particulièrement apprécié la qualité des retours et leurs commentaires constructifs, malgré le peu de temps disponible et la longueur de ce manuscrit.

Je souhaite de même remercier les autres membres de mon jury de thèse, Natalia Neverova et Irène Gaillard de leur attention et pour les discussions que nous avons pu avoir lors de la soutenance.

Je remercie Vincent Charvillat, mon directeur de thèse, qui a appuyé ma candidature puis m'a fait confiance tout au long de la thèse. Je veux aussi remercier mes deux co-encadrants, Axel Carlier et Sandrine Mouysset pour leur disponibilité, patience et leur bienveillance, ainsi que pour leur soutien tout au long de mon doctorat et pour toutes ces discussions qui m'ont permis de prendre du recul afin de trouver des idées pertinentes pour résoudre les problèmes posés. Je remercie Philippe Milles, président de la société Ebhys, pour la confiance et l'autonomie qu'il m'a laissées dans la réalisation de mes travaux de recherche, mais aussi Alain Fluhr, pour les efforts de compréhension réalisés afin de pouvoir suivre mes travaux et les discussions, questions et remarques intéressantes qui en ont découlé.

Je souhaite remercier tous les permanents de l'équipe Reva. Je remercie Géraldine, qui m'a accueilli, encadré et intégré à l'équipe lors de mon projet de fin d'étude, Sylvie pour les nombreuses discussions que nous avons eues, les cours que nous avons organisés ensemble et pour m'avoir aidé à répéter la soutenance, Pierre pour tous les conseils concernant la course à pied, Jean-Denis pour l'organisation des rassemblements dans le Lot et Simone pour ses excellents tiramisus. Merci à tous du temps et de l'énergie que vous dépensez pour faire vivre l'équipe.

Je remercie par ailleurs tous les membres non permanents que j'ai pu côtoyer : Yvain, Bastien, Vincent, Julien, Thomas, Sonia, Matthieu, Jean, Damien, Clément, Arthur, Thierry, Julien (le deuxième), Paul, Rémi et Marie. Merci beaucoup pour tous ces moments informels, tels que ces repas et ces soirées que nous avons pu partager, ainsi que pour l'ambiance chaleureuse de l'équipe qui leur doit beaucoup. Je remercie particulièrement Thomas pour

l'aide qu'il m'a apportée d'un point de vue logistique lors de la soutenance, et je suis désolé de l'avoir oublié lors des remerciements à la fin de celle-ci.

Je remercie aussi les stagiaires et les groupes d'étudiants que j'ai eu la chance d'encadrer pour l'intérêt qu'ils ont pu montrer concernant mes travaux et pour tout ce qu'ils ont pu y apporter.

Merci à Alan, Aline, Arthur, Christophe, Frédéric C., Frédéric D., Stéphane et Virginie, mes collègues chez Ebhys, pour l'accueil qu'ils m'ont réservé au sein de leur entreprise et pour tous les services qu'ils m'ont rendus lorsque j'étais au Thor.

En plus d'Irène, examinatrice de ma thèse, je remercie Vanina, Amélie et Mélanie, ergonomes au CERTOP, avec qui j'ai pu collaborer afin d'apporter une dimension humaine à mes travaux.

Je remercie Anabelle, Isabelle, Muriel et Sylvie pour toute la gestion administrative liée à ma thèse.

Plus personnellement, je tiens à remercier tous mes amis, éparpillés à Toulouse, Paris, Londres, Grenoble, dans le Vercors ou ailleurs, pour le soutien qu'ils m'ont apporté et pour tous les bons moments. Je remercie aussi tous les membres de l'équipe de bénévoles des Restaurants du Cœur pour la Distribution du Grand Rond du vendredi, grâce à qui je passe mes vendredis soirs dans une ambiance chaleureuse et décontractée.

Enfin, je remercie ma famille, pour m'avoir accompagné, soutenu et pour l'intérêt qu'ils ont apporté à mes travaux lors des nombreux repas et fêtes de famille où j'ai eu l'occasion de leur en parler. Plus particulièrement, je remercie mes parents, mon frère et ma sœur pour l'accompagnement et le soutien quotidiens. Pour finir, je remercie profondément Mathilde, pour la patience et le soutien indéfectible dont elle a fait preuve pendant toute la durée de cette thèse.

Table des matières

Table des figures

Liste des tableaux

Introduction

Durant les dernières années, les données ainsi que leur gestion et leur traitement ont pris une importance sans précédent dans notre société. Le volume de données créées chaque année augmente de manière exponentielle, et elles sont une ressource cruciale pour les entreprises et les États. Dans le même temps, les récentes avancées techniques et technologiques, avec des objets de plus en plus connectés, donnent une place de plus en plus importante à l'intelligence artificielle dans notre vie de tous les jours.

Ces deux éléments sont intimement liés, et permettent le développement ou l'amélioration de nombreux services et outils. L'acquisition de données, souvent personnelles, et leur traitement par des algorithmes intelligents permet par exemple la suggestion de contenu intéressant à lire ou à regarder, mais aussi la mise en place de publicité ciblée, ou bien le développement de dispositifs de vidéosurveillance, tout en permettant de contenir la propagation d'épidémies. Aussi, l'acquisition puis le traitement de nos données par des intelligences artificielles devient un sujet de plus en plus sensible : si le citoyen tire un intérêt évident de certaines applications, il est aujourd'hui très difficile de savoir quelles données sont enregistrées, et dans quel but elles sont utilisées.

Le monde de l'industrie n'est pas épargné par cette tendance. En effet, de plus en plus d'entreprises investissent dans le développement et la mise en place de dispositifs d'acquisition et de traitement des données, avec plusieurs objectifs en ligne de mire, tels que la facilitation de la gestion du fonctionnement d'une usine, la prédiction des besoins de maintenance des machines, une amélioration de la qualité ou bien une amélioration du service après-vente. Cependant, la collecte et l'analyse des données dans l'industrie reste difficile à réaliser, à cause des nombreuses contraintes imposées par le contexte industriel.

1 Sujet de la thèse

C'est dans ce contexte que prennent place les recherches présentées dans cette thèse. La société Ebhys, ingénieriste ensemblier d'unités de valorisation des déchets, souhaite développer un outil d'analyse de l'activité d'un centre de tri. Ce dispositif doit en particulier permettre d'obtenir des informations sur le fonctionnement du centre, utiles à l'exploitant pour le gérer au mieux, et à la société Ebhys, afin d'avoir un retour sur la qualité de la conception. L'objectif, in fine, est donc d'acquérir des données et de les traiter de manière

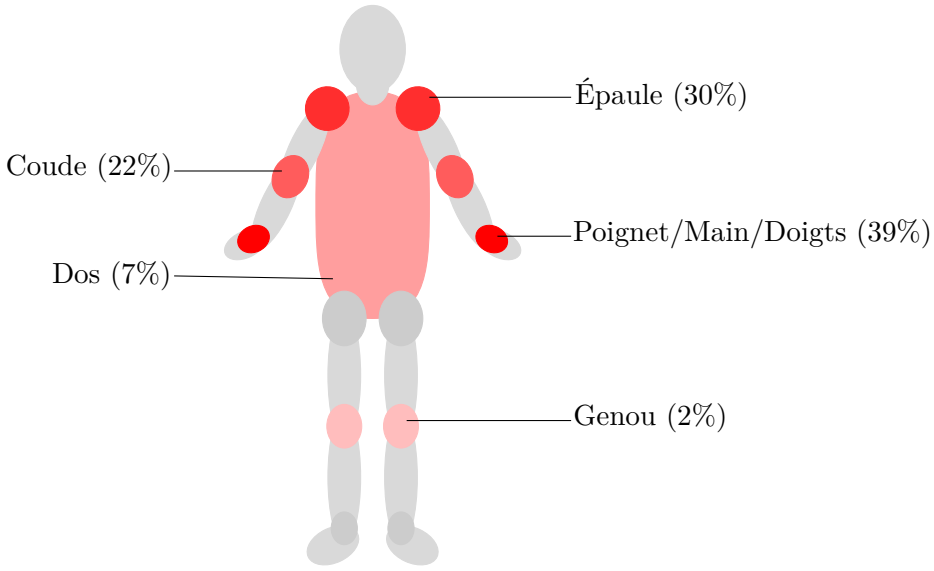


FIGURE 1 – Les troubles de musculo-squelettiques (TMS) forment un groupe de pathologies qui représente la première maladie professionnelle en France et dans les pays industrialisés. Ils touchent surtout les membres supérieurs, comme illustré par cette figure. Source : Assurance Maladie, chiffres 2017.

intelligente pour optimiser l'exploitation du centre et les conceptions futures. Le développement d'un tel dispositif peut cependant paraître ambigu : si l'objectif est de développer une plateforme de prévention pour la santé et l'amélioration des conditions de travail des opérateurs, il pourrait aussi être perçu ou se transformer en outil de surveillance des travailleurs, en fonction de l'usage qui en est fait.

Nos recherches se concentrent sur une partie importante de ce dispositif : l'analyse de l'activité des opérateurs humains dans un centre de tri. Le but de cette analyse est d'effectuer des mesures ergonomiques afin de prévenir l'apparition des troubles musculo-squelettiques, dont la répartition est donnée en Figure 1. En effet, le métier d'agent de tri est un métier pénible, car l'opérateur effectue un grand nombre d'actions de manière répétitive, réalisant ainsi jusqu'à quarante actions par membre supérieur par minute, d'après la norme *NF X35—702*. Le développement, sur le long terme, de troubles musculo-squelettiques est donc fréquent chez les agents de tri. L'analyse de l'activité est toutefois délicate, car l'humain dispose d'une grande liberté pour effectuer une tâche, au contraire d'un automate, souvent déterministe. En outre, le contexte industriel, non contrôlé, augmente la complexité de l'acquisition des données et leur traitement : le déploiement du dispositif en environnement industriel introduit par exemple des contraintes telles que la présence de vibrations, la variabilité des conditions d'éclairage ou le port de vêtements réfléchissants. Les données acquises seront donc probablement imprécises, et il sera difficile de les annoter, rendant l'utilisation d'algorithmes d'intelligence artificielle, basés sur l'apprentissage automatique pour l'extraction d'information, difficile. Ainsi, l'analyse de

l'activité des opérateurs humains en centre de tri est un véritable défi scientifique, mais aussi un challenge de recherche contenant une problématique d'acceptabilité du dispositif par les opérateurs.

Au-delà de leur utilité applicative, les travaux réalisés dans cette thèse seront donc des travaux qui devront répondre à la question suivante :

Comment construire une information pertinente à partir de données imprécises ou incomplètes ?

Il est nécessaire de définir plusieurs termes relatifs à cette problématique.

Tout d'abord, qu'est-ce qu'une information pertinente ? Il s'agit dans un premier temps d'une information qui ait un sens et qui soit facilement interprétable par un utilisateur. Dans notre cas, une information pertinente est donc par exemple un indicateur simple de pénibilité. Dans un second temps, une information pertinente est une information qui soit en accord avec les faits observés. Ainsi, cette information doit être le fruit d'une interprétation correcte des données qui en sont la source.

Ensuite, qu'est-ce qu'une donnée imprécise ou incomplète ? Une donnée imprécise est une donnée bruitée, c'est-à-dire une donnée qui ne reflète pas entièrement la réalité. Dans notre cas, le bruit vient des contraintes de l'environnement industriel. Une donnée incomplète est une donnée partielle, c'est-à-dire une donnée qui ne peut pas être utilisée seule. Dans notre cas, l'absence d'annotation pour l'entraînement d'algorithmes d'intelligence artificielle basés sur l'apprentissage automatique rend nos données incomplètes.

Ainsi, l'objectif de cette thèse sera de définir des procédés d'apprentissage qui permettront d'extraire au mieux une information, à partir de données bruitées, et sans annotation, avec en ligne de mire une application industrielle à l'analyse de l'activité d'opérateurs humains dans un centre de tri de déchets. Le problème à résoudre est donc à la fois technique et technologique.

2 Plan de la thèse

Pour répondre à ce problème, ce manuscrit de thèse est organisé en plusieurs chapitres, dont l'articulation est illustrée en Figure 2.

Premièrement, dans le Chapitre 1, nous exposons les informations préliminaires qui permettent de comprendre les enjeux de la thèse. Nous analysons le contexte industriel dans lequel s'est déroulée cette thèse, puis nous décrivons en détail les troubles musculosquelettiques. Nous introduisons la norme volontaire *NF X35—702* et les contraintes imposées sur le projet par l'environnement industriel, et nous présentons le type de capteur choisi pour acquérir les données à traiter.

Dans le Chapitre 2, nous décrivons la problématique qui se dégage suite à l'analyse du contexte dans lequel s'effectuent les travaux. Nous exposons le procédé de construction

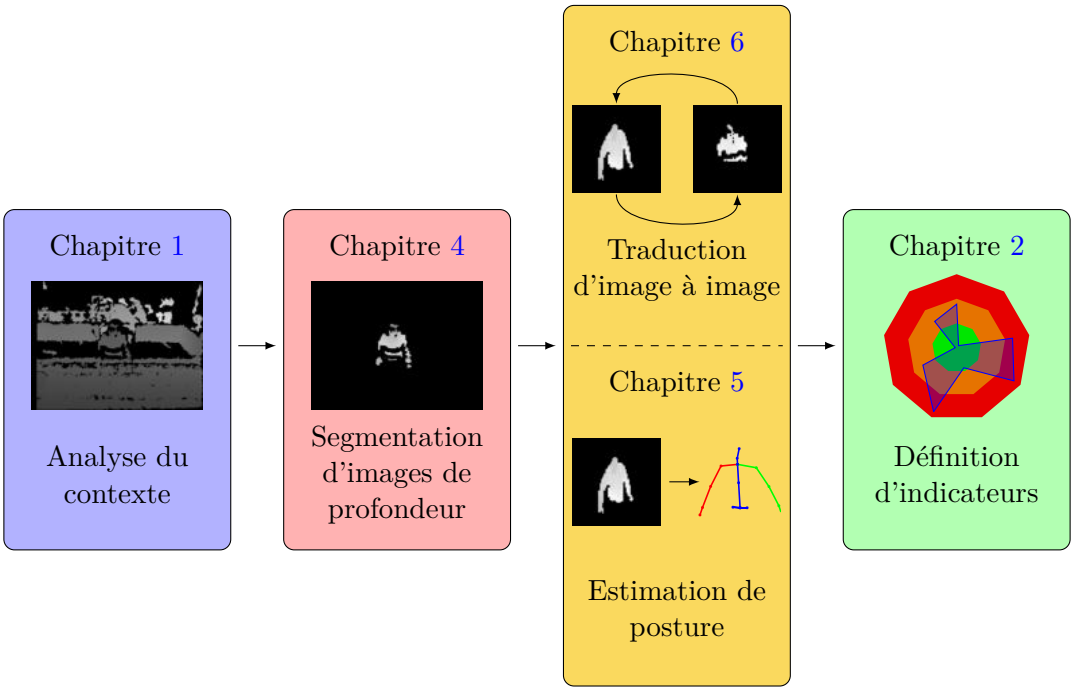


FIGURE 2 – Organisation des différents chapitres de la thèse. Le Chapitre 1 présente le contexte de la thèse. Le Chapitre 2 définit les indicateurs à mesurer ainsi que la problématique de la thèse. Le Chapitre 3 propose un état de l'art utile à la compréhension des chapitres suivants. Le Chapitre 4 introduit notre approche pour la segmentation faiblement supervisée d'images de profondeur. Le Chapitre 5 présente un procédé de génération d'images de synthèse et d'estimation de la posture sur des images de profondeur annotées. Enfin, le Chapitre 6 décrit notre approche de traduction d'image à image, nous permettant d'estimer la posture sur des images de profondeur réelles.

des indicateurs de pénibilité ainsi que le processus à mettre en place pour mesurer ces indicateurs.

Ensuite, dans le Chapitre 3, nous passons en revue l'état de l'art en matière de segmentation d'image et d'estimation de posture. Ce chapitre présente les approches traditionnelles, n'ayant pas recours à l'apprentissage profond. Nous décrivons par ailleurs différents types d'apprentissage, puis nous introduisons le concept d'apprentissage profond, en présentant les réseaux de neurones ainsi que des mesures permettant d'évaluer leur qualité.

Dans le Chapitre 4, nous présentons notre première contribution : une approche de segmentation d'images de profondeur par apprentissage faiblement supervisé. Nous décrivons tout d'abord un premier algorithme automatique de segmentation d'image de profondeur, qui permet de construire un ensemble de données de qualité variable. Deuxièmement, nous exposons une nouvelle technique d'entraînement d'un modèle d'apprentissage profond, où l'ensemble d'apprentissage est itérativement mis à jour pour filtrer les données aberrantes. Nous évaluons et démontrons l'intérêt de cette approche à l'aide d'expériences réalisées

sur nos données.

Dans le Chapitre 5, nous exposons nos deuxième et troisième contributions. Nous définissons un procédé de génération de données de synthèse, puis nous décrivons les travaux qui nous ont permis de construire un réseaux de neurones optimal et compact pour l'estimation de posture en 3D sur des images de profondeur de synthèse.

Après avoir construit un modèle performant pour l'estimation de posture 3D sur des images de synthèse, nous présentons dans le Chapitre 6 une des contributions les plus importantes de cette thèse. Dans une première partie, nous décrivons notre modèle de traduction d'image à image qui permet de transformer des données de synthèse en données réalistes. Ensuite, nous entraînons un modèle de réseau de neurones en utilisant les images de profondeur réalistes. Les expériences réalisées montrent que cette approche est efficace pour apprendre à un modèle à traiter des données pour lesquelles on ne possède pas d'annotations.

Chapitre 1

Contexte

Sommaire

Le métier d'agent de tri, ou valoriste, induit une répétitivité des mouvements qui peut à long terme comporter des risques de développement de certaines pathologies (comme par exemple les troubles musculo-squelettiques), à l'instar de nombreux autres métiers.

En effet, les troubles musculo-squelettiques sont les maladies professionnelles les plus courantes dans les pays développés. Ils affectent généralement les tissus mous (muscles, articulations ou nerfs) et sont favorisés par certains facteurs comme la posture de l'opérateur de tri ou la répétitivité de la tâche à effectuer. La société Ebhys souhaite donc développer des composants informatiques et métrologiques permettant la prévention et l'appréhension de ces pathologies.

Afin d'améliorer les conditions de travail des trieurs, la norme volontaire *NF X35—702* a vu le jour en 2015. Cette norme définit les actions techniques requises pour la réalisation des opérations effectuées au cours du travail, quantifie un nombre acceptable d'actions techniques et prend en compte la zone d'action de l'opérateur, qui influe sur le confort de sa position et donc la pénibilité de l'action. Impossible à vérifier, cette norme est volontaire.

La pression constante des exploitants de centre de tri et des collectivités pour améliorer les conditions de travail des trieurs sur les installations industrielles conduit à envisager un nouveau développement en matière d'ergonomie sur les postes de tri.

Néanmoins, la prise en compte de la norme *NF X35—702* nécessite un suivi constant des opérateurs afin d'estimer le nombre d'actions techniques réalisées et la pénibilité de leur position. Pour ce faire, la société Ebhys cherche à mettre en place un système automatique de mesure de l'activité d'opérateurs humains dans un centre de tri, qui constitue l'objectif de cette thèse.

Ce chapitre présente donc le contexte qui guide très fortement la réalisation de cette thèse. Dans une première partie, nous présentons le centre du tri et la place de l'agent de tri dans son fonctionnement (Section 1.1). Ensuite, nous décrivons les troubles musculo-squelettiques (TMS) à travers leurs symptômes, leurs causes et leurs conséquences (Section 1.2). Nous introduisons par la suite la norme *NF X35—702* (Section 1.3), puis les contraintes imposées sur le projet par l'environnement industriel, l'entreprise et le capteur utilisé (Section 1.4). Enfin, nous présentons le type de capteur sélectionné pour mener à bien ce projet (Section 1.5).

1.1 Le centre de tri des déchets

Le passage par le centre de tri des déchets est une étape essentielle dans le cycle du recyclage et de la valorisation des déchets. En effet, le centre de tri réceptionne les déchets amenés par les camions de collecte sélective, les trie puis les revend au recycleur. Cette étape de tri permet de purifier les flux de matière afin qu'elle serve de base à la production de nouveaux objets. Le centre de tri a pour objectif de diviser les déchets en fractions, en séparant les différentes catégories de matière. Le centre de tri fait donc le lien entre la

collecte plus ou moins sélective des déchets et le recycleur, pour lequel il fournit la matière première nécessaire.

L'agence de l'environnement et de la maîtrise de l'énergie (ADEME) estime que le nombre de centres de tri en activité en France est d'environ 450, toutes activités confondues, avec 176 centres de tri dédiés aux déchets issus de la collecte sélective en 2018 [Citeo et Adelphe, 2018].

1.1.1 Fonctionnement d'un centre de tri

Il est possible de mettre en place différentes techniques pour trier les déchets. Le tri est aujourd'hui très largement automatisé grâce à l'utilisation de machines de plus en plus perfectionnées :

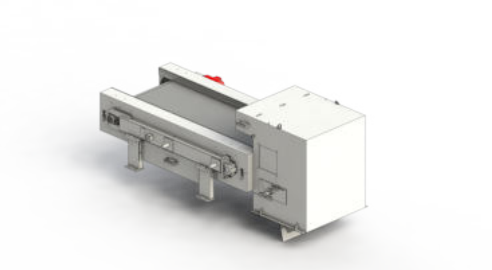
- les séparateurs (ou cribles) balistiques qui séparent la matière en trois fractions différentes : les plus petits déchets sont criblés par les grilles du séparateur alors que les corps légers et plats comme les films se retrouvent dans la partie supérieure du séparateur, et les corps lourds, tridimensionnels, roulent et se dirigent vers le bas de la machine ;
- les séparateurs à courant de Foucault qui permettent d'extraire l'aluminium du reste du flux ;
- les trieurs optiques qui affinent le pré-tri effectué par les séparateurs balistiques : fonctionnant sur le principe de la spectrométrie infrarouge, les trieurs optiques reconnaissent la matière circulante grâce à un spectromètre et séparent les différents types de matière à l'aide de projections d'air comprimé ;
- les robots trieurs : récemment introduits sur le marché (en 2017), ces robots utilisent l'intelligence artificielle et une approche basée sur l'apprentissage profond pour effectuer un tri très fin de la matière sur la table. Équipées d'un bras articulé et capables de trier jusqu'à 3600 objets par heure (un objet par seconde), ces machines ont pour objectif de remplacer peu à peu les agents de tri. Elles ont cependant encore quelques défauts, comme par exemple le besoin que la matière soit très étalée sur la table afin de pouvoir correctement identifier les objets.

Des illustrations de ces équipements sont données en Figure 1.1. Toutes ces machines sont reliées entre elles par le biais de convoyeurs (tapis roulants), sur lesquels transite la matière tout au long de son parcours sur la chaîne de tri.

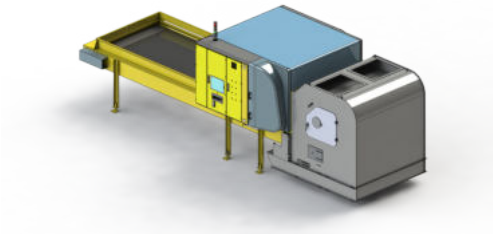
Cette automatisation du tri permet d'augmenter le volume de matière triée tout en réalisant un tri plus fin avec un nombre de matières premières à recycler plus important.

La présence d'opérateurs humains reste néanmoins encore nécessaire pour corriger les erreurs de tri effectuées par les machines, et le travail de l'agent de tri se transforme peu à peu en une mission de contrôle de la qualité des flux sortants à la fin de la chaîne de tri.

Séparateur balistique



Séparateur à courant de Foucault



Trieur optique

Robot trieur

FIGURE 1.1 – Images illustratives des équipements de tri automatique présents dans les centres de tri : le séparateur balistique, le séparateur à courant de Foucault, le trieur optique et le robot trieur.

1.1.2 L'opérateur humain dans un centre de tri

L'agent de tri, aussi appelé plus simplement trieur ou valoriste, est un des opérateurs humains les plus importants de la structure. En effet, sa mission est d'intervenir sur la chaîne de tri pour garantir une séparation optimale des différents types de matière. Plus particulièrement, son rôle est d'identifier la nature des déchets pour ensuite séparer les principales catégories de matières recyclables entre elles, mais aussi d'ôter de la ligne de tri les refus, composés des matières non recyclables.

Le travail de l'agent de tri se déroule dans une cabine de tri.

1.1.3 La cabine de tri

La cabine de tri est définie comme le local dans lequel s'effectue l'activité de tri manuel, c'est-à-dire que c'est le lieu dans lequel les agents de tri réalisent leur mission. Un exemple illustratif de cabine de tri est donné en Figure 1.2.

Le nombre de cabines de tri au sein d'une même unité de tri est variable et dépend souvent de différents critères qui tiennent compte de la gestion de l'activité humaine (visibilité des flux de matière, quantité de déchets, rotation et polyvalence des agents, etc.), de la gestion de l'environnement immédiat de la cabine (centralisation des accès, circulation entre les zones de tri, etc.) et de la gestion de l'ambiance physique (harmonisation de la température, ventilation, bruit, lumière naturelle, éclairement, vision extérieure, etc.).

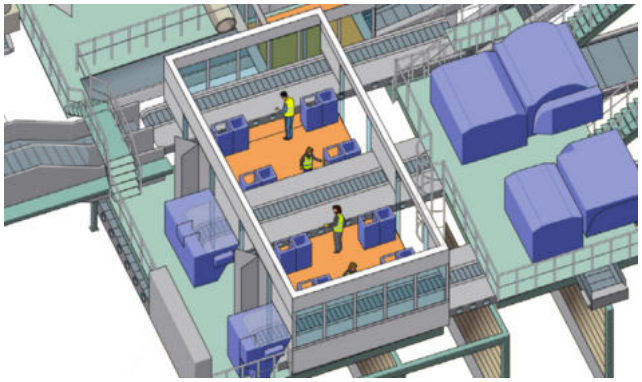


FIGURE 1.2 – Exemple illustratif de cabine de tri à l'intérieur du centre de tri. La cabine de tri est le local au sein duquel s'effectue le tri réalisé par les agents de tri.

Ainsi, les centres de tri contiennent parfois plusieurs cabines de tri pour le tri des différents flux de matière (pré-tri, corps creux, corps plats, déchets légers, déchets lourds, etc.), mais la norme *NF X35—702* pour la conception de la cabine de tri manuel préconise de regrouper l'activité de tri dans une unique cabine de tri multi-déchets [AFNOR, 2015].

La conception de la cabine de tri doit respecter les règles et les recommandations énoncées par la norme *NF X35—702*. En particulier, les règles pouvant impacter le développement sont les suivantes :

- la cabine de tri doit être isolée thermiquement du reste du centre : la température dans la cabine de tri doit être comprise entre 18°C et 23°C ;
- la cabine de tri doit être isolée phoniquement du reste de l'unité de tri : le niveau sonore au poste de travail ne doit pas excéder 75dB ;
- les cloisons de la cabine de tri doivent être équipées de surfaces transparentes et permettre une visibilité sur l'extérieur du bâtiment ;
- la totalité des parois de la cabine doit pouvoir faire l'objet d'un nettoyage humide ;
- les vibrations créées par les équipements de tri automatique ne doivent pas être transmises à la cabine de tri.

Les conséquences de ces contraintes sur le projet sont expliquées plus en détail en Sections 1.3 et 1.4.

Les équipements de tri automatique, tels que ceux décrits précédemment (voir Section 1.1.1), sont généralement situés en amont sur la ligne de tri, sauf pour les tables de pré-tri.

En règle générale, une cabine de tri regroupe plusieurs flux de matières distincts, c'est-à-dire plusieurs lignes. Ces flux sont traités sur des tables différentes.

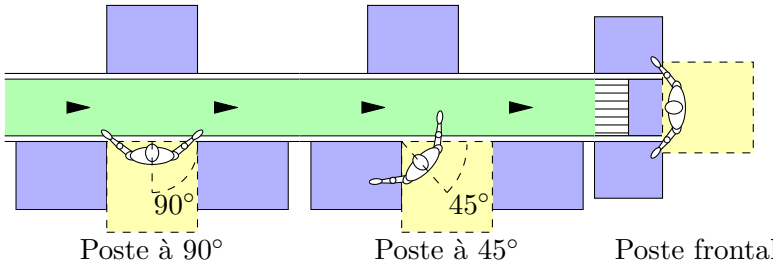


FIGURE 1.3 – Positionnements possibles des postes (en jaune) et agents de tri sur la chaine de tri, le long d'une table de tri unilatéral (en vert). La partie hachurée représente un plan incliné en fin de table et les exutoires potentiels sont colorés en bleu.

1.1.4 La table de tri et le poste de tri

La table de tri est le plan de travail de l'agent de tri. Elle est horizontale, mais ses dimensions dépendent de la taille des objets à trier (voir Section 1.3.1). La table contient un tapis roulant appelé convoyeur ou bande.

La table de tri est divisée en postes de tri, situés le long de la table (postes à 45°et 90°) ou au bout de la table (poste frontal). Le tri peut être unilatéral, si tous les postes de tri sont situés du même côté de la table, ou bien bilatéral, si des postes de tri sont positionnés des deux côtés de la table. Ces postes de tri sont entourés d'exutoires, c'est-à-dire des équipements d'évacuation des déchets triés qui peuvent être des goulottes ou des convoyeurs, pour déposer, faire glisser ou lancer la matière prélevée sur la table. Il existe deux types de tris : le tri négatif, où les opérateurs retirent de la table les déchets indésirables pour ne conserver que la fraction valorisable et le tri positif, consistant à prélever du flux une fraction de matière valorisable. Un schéma illustratif de table de tri et des différents types de postes de tri est donné en Figure 1.3.

1.2 Les troubles musculo-squelettiques

Les troubles musculo-squelettiques (TMS), aussi appelés "*Repetitive Strain Injuries*" (RSI) aux États-Unis, "Lésions attribuables au travail répétitif" (LATR) au Québec ou "Lésions par Efforts Répétitifs" (LER) au Brésil [Delalande-Danet et al., 2015], sont un groupe de pathologies et constituent la première maladie professionnelle en France et dans les pays industrialisés en général [Punnett and Wegman, 2004]. En effet, l'activité professionnelle peut jouer un rôle dans l'apparition, le maintien et l'aggravation de ces troubles. Ces troubles touchent en général les tissus mous, tels que les muscles, les tendons et les nerfs. Ils se traduisent principalement par des douleurs périarticulaires et une gêne fonctionnelle souvent quotidiennes, mais aussi par de la maladresse, une perte de force, une raideur ou de la fatigue [Ha and Roquelaure, 2010].

En 2015, les TMS représentaient plus de 87% des maladies professionnelles. Les TMS peuvent devenir irréversibles et entraîner un handicap durable, s'ils ne sont pas diagnostiqués et pris en charge rapidement.

1.2.1 Réglementation

Il n'existe à l'heure actuelle pas de réglementation spécifique à propos des TMS en matière de sécurité et de santé au travail. Cependant, l'employeur a pour obligation de préserver la santé physique et mentale de ses employés (d'après l'article L. 4121—7021 du Code du travail). Ainsi, il doit prendre les mesures de prévention nécessaires pour éviter toute maladie ou accident professionnel [INRS, 2015].

Ces pathologies ont par ailleurs été identifiées comme un problème majeur de santé publique au début des années 2000 [Hatzfeld, 2006].

Certaines de ces pathologies sont reconnues comme des maladies professionnelles indemnisables par la législation française [Delalande-Danet et al., 2015]. Elles sont données sous forme de tableaux regroupant les maladies reconnues [Ministère de la santé, 2020], les délais de prises en charge et les travaux pouvant les provoquer. Le principal tableau décrivant les TMS est le tableau n°57 [Ministère de la santé, 2012] : il décrit les affections périarticulaires provoquées par certains gestes et postures de travail concernant l'épaule, le coude, le poignet, la main, le doigt, le genou, la cheville et le pied qui représentaient environ 80% des pathologies professionnelles en 2014 [Ministère du travail, 2014], et environ 92% des TMS diagnostiqués.

D'autres tableaux décrivent des pathologies qui peuvent être assimilées aux TMS :

— le tableau n°69 : affections provoquées par les vibrations et chocs transmis par certaines machines-outils, outils et objets et par les chocs itératifs du talon de la main sur des éléments fixes ;
— le tableau n°79 : lésions chroniques du ménisque ;
— le tableau n°97 : affections chroniques du rachis lombaire provoquées par des vibrations de basses et moyennes fréquences transmises au corps entier ;
— le tableau n°98 : affections chroniques du rachis lombaire provoquées par la manutention manuelle de charges lourdes

Les causes de ces troubles sont cependant très complexes à mesurer, la prévention de ceux-ci ne fera donc dans pas l'objet de cette thèse.

1.2.2 Anatomie des membres supérieurs

Pour connaître la localisation des TMS, il est important de bien comprendre la structure des membres supérieurs du corps humain (os, articulations, nerfs, muscles, etc.).

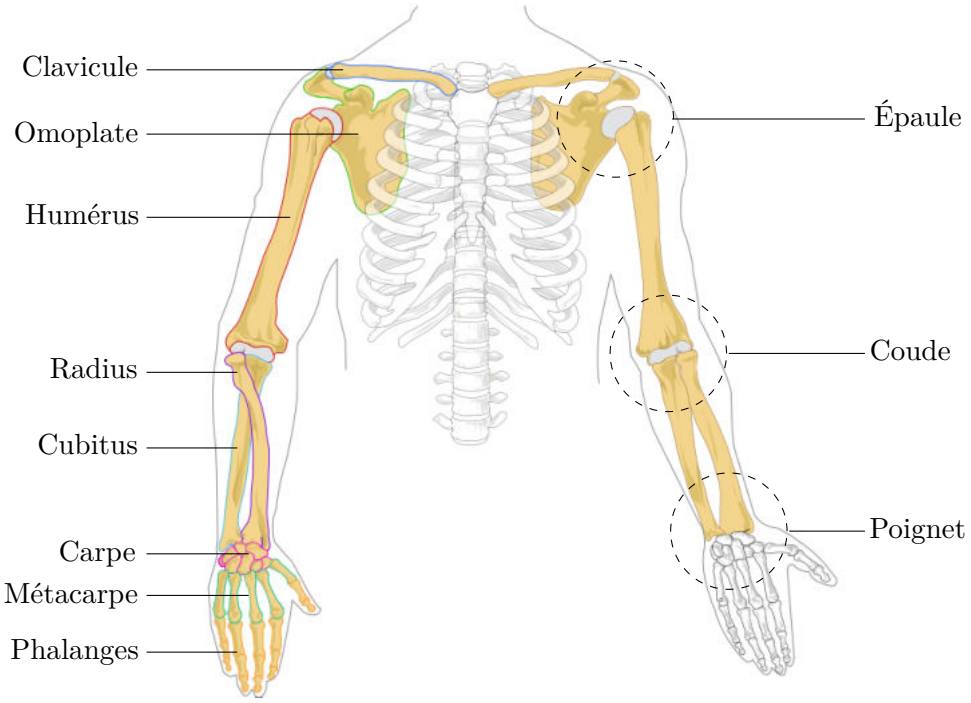


FIGURE 1.4 – Les os et les articulations des membres supérieurs. Image issue de [Wikipédia, 2019].

Le squelette du membre supérieur

L'épaule est composée de trois os : l'omoplate, la clavicule et la tête de l'humérus. Le bras est composé d'un os (l'humérus), tandis que l'avant-bras est composé de deux os : le cubitus et le radius. Enfin, la main est constituée des os du carpe (huit os), des métacarpiens (cinq os) et des phalanges (deux pour le pouce et trois pour les autres doigts). Le squelette des membres supérieurs est donné en Figure 1.4.

Les articulations

Les articulations sont constituées des surfaces articulaires des os revêtues de cartilage et des structures assurant leur liaison, comme la capsule articulaire et les ligaments.

La capsule articulaire est composée de deux couches : la membrane synoviale (interne) et la membrane fibreuse (externe). La membrane synoviale contient des fibres élastiques, des vaisseaux et des nerfs. La cavité articulaire contient la synovie. La synovie est un lubrifiant qui s'écoule entre les surfaces articulaires lorsque l'articulation est en mouvement afin de faciliter celui-ci et en absorber les chocs. Elle est réabsorbée par la membrane synoviale lorsque l'articulation est au repos.

Les ligaments sont les liens entre les structures osseuses formant l'articulation. Ils en assurent la stabilité passive et sont généralement inclus dans la capsule articulaire.

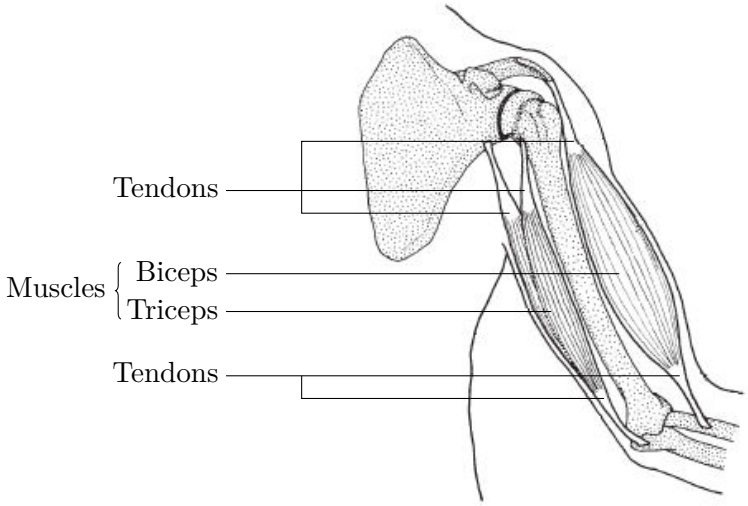


FIGURE 1.5 – Exemple de muscles et de tendons des membres supérieurs avec les muscles de l'avant-bras : le biceps et le triceps.

Les bourses séreuses sont d'autres éléments périarticulaires. Ce sont des « coussins » remplis de liquide synovial qui sont situés au niveau des grosses articulations (coude, genou, cheville, etc.).

Les muscles

Les muscles sont composés de fibres musculaires qui peuvent se contracter ou s'allonger en fonction de la charge physique. Les phases de contraction alternent avec les phases de relâchement. La contraction est le raccourcissement du muscle qui permet la mise en mouvement de l'os sur lequel il s'insère. Ils sont reliés à l'os par le biais de tendons. Une illustration des muscles du bras est donnée en Figure 1.5.

Dans le travail statique, la contraction musculaire est maintenue dans le temps sans modification de la longueur musculaire. Dans ce cas, le muscle présente seulement des variations de force.

Le membre supérieur est constitué de plus de quarante muscles.

Les tendons

Les tendons constituent le prolongement du muscle et assurent son insertion sur l'os : ils font le lien entre le squelette et les muscles. Ce sont des éléments fibro-élastiques qui ont une longueur variant de quelques millimètres à plusieurs centimètres. Une gaine synoviale protège certains tendons des frictions contre les autres éléments des articulations.

Les nerfs

Les nerfs assurent la conduction des ordres moteurs ou des informations sensorielles. Les informations qu'ils transmettent sont codées sous forme de potentiels d'action. Les nerfs sont vascularisés. Les racines nerveuses qui émergent entre les vertèbres cervicales (au niveau du cou) se rassemblent pour former le plexus brachial. Celui-ci donne naissance à l'ensemble des nerfs du membre supérieur [INRS, 2011a].

Le nerf ulnaire (ou cubital) et le nerf médian sont les principaux nerfs du membre supérieur. Ils innervent tous les deux des muscles de l'avant bras et de la main. Au niveau du poignet, on trouve le nerf médian dans le canal carpien (situé au niveau du carpe). À travers ce canal inextensible passent aussi neuf tendons fléchisseurs des doigts.

1.2.3 Les principaux troubles musculo-squelettiques

Les TMS touchent les tissus mous périarticulaires. Ainsi, ils peuvent toucher les muscles, les tendons, les ligaments et les nerfs, mais aussi les vaisseaux sanguins, les bourses séreuses ou encore les cartilages [Ha and Roquelaure, 2010].

Les principaux TMS qui touchent les valoristes sont les suivants :

— l'**hygroma** ou **bursite** : elle correspond à l'inflammation d'une bourse séreuse. Cette inflammation peut être aggravée par le mouvement des tendons et des muscles proches de la bourse. Elle peut être localisée à l'épaule, au coude (en cas de sur-utilisation de ces articulations) ou au genou (à cause d'une posture prolongée) ;

— les **lombalgies** : c'est le terme médical utilisé pour désigner les douleurs du bas du dos, dans la région des vertèbres lombaires. Une lombalgie peut survenir à la suite d'un effort inhabituel ou de l'accumulation de nombreux efforts ayant par exemple engendrés des micro-lésions ou bien être due à la vieillesse ;

— les **myalgies** : ce sont des douleurs musculaires. Le terme englobe toutes les douleurs musculaires du corps charnu du muscle. C'est en général la conséquence d'un excès de tonus des muscles entraînant généralement une raideur (hypertonie musculaire) ;

— le **syndrome du tunnel cubital ou ulnaire** : localisé au niveau du coude ou de la main (appelé dans ce cas syndrome de la loge de Guyon), il s'agit d'une compression due à un excès de pression sur le nerf cubital. Cette compression entraîne des engourdissements et une perte de fonction de la main, car les influx nerveux ont du mal à passer ;

— le **syndrome du canal carpien** : c'est une affection fréquente résultant d'une compression du nerf médian au niveau du canal carpien. Il a sensiblement les mêmes conséquences que le syndrome du tunnel cubital ;

— les **tendinites** : elles correspondent à une inflammation d'un ou plusieurs tendons. Elles peuvent toucher l'épaule (tendinite de la coiffe des rotateurs), le coude (épicondylites latérale ou médiale) ainsi que le poignet et la main (tendinites des fléchisseurs

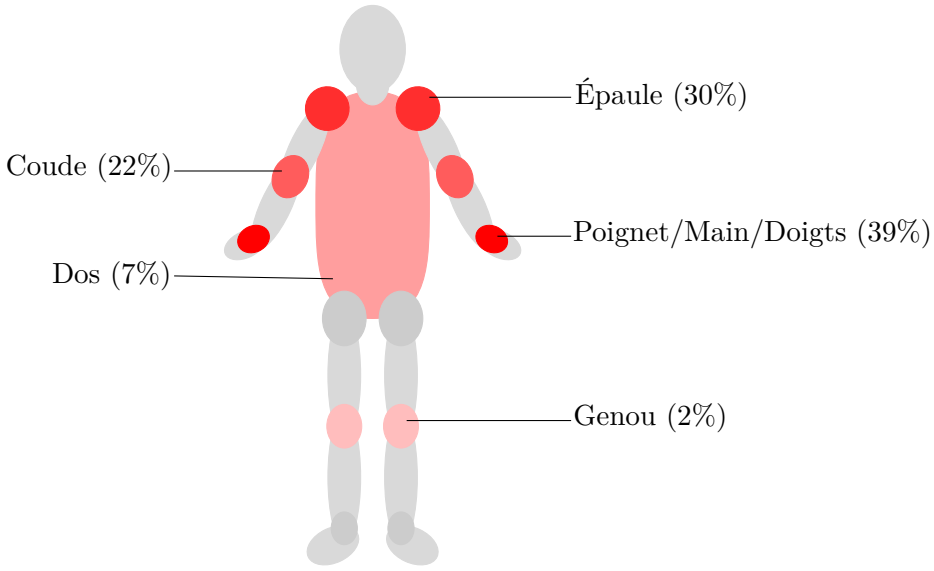


FIGURE 1.6 – La répartition des syndromes de musculo-squelettiques (TMS) par localisation. Les TMS atteignent dans une très large majorité des cas les membres supérieurs. Source : Assurance Maladie, chiffres 2017.

et des extenseurs de la main et des doigts) ;

— les **ténosynovites** : localisées au niveau du poignet (maladie de De Quervain), elles sont caractérisées par l'inflammation et le gonflement du tendon, provoquant une irritation et une inflammation de la gaine synoviale qui entoure le tendon.

Le tableau 1.1 détaille les différentes TMS pouvant atteindre chacune des articulations, ainsi que leur(s) tableau(x) de référence [Ministère de la santé, 2020]. Les TMS les plus souvent diagnostiqués sont les TMS de l'épaule, suivis par le syndrome du canal carpien et les TMS du coude [Brière et al., 2015]. La Figure 1.6 illustre la répartition des syndromes de TMS par localisation.

1.2.4 Facteurs de risque

Les TMS sont des pathologies professionnelles multifactorielles liées à des contraintes biomécaniques, organisationnelles, individuelles, environnementales et psychosociales. En général, c'est l'interaction entre plusieurs de ces facteurs de risque qui est à l'origine du risque de TMS. Les troubles musculo-squelettiques du membre supérieur (abrégés TMS-MS) ne suivent pas un modèle mécaniste (comme pour le bruit et la surdité par exemple), mais plutôt un modèle probabiliste où différents facteurs augmentent la probabilité d'apparition des maladies [INRS, 2011a, Delalande-Danet et al., 2015].

L'influence de chacun des facteurs dans la survenue des TMS dépend de la fréquence d'exposition à ces facteurs, de l'intensité de cette exposition et également de la durée d'exposition. En effet, les TMS sont les conséquences d'un abus répété d'une structure, par

Articulation	Type de syndrome	Tableau de référence
Coude	Compression du nerf cubital	n°57
	Arthrose du coude	n°69
	Hygroma aigu du coude	n°57
	Épitrochléite	n°57
	Épycondilite	n°57
Main/Poignet	Troubles angloneurotiques de la main	n°69
	Maladie de Klenböck	n°69
	Maladie de Köhler	n°69
	Atteinte vasculaire cubito-palmaire	n°69
	Syndrome du canal carpien	n°57
	Syndrome de la loge de Guyon	n°57
	Ténosytovite	n°57
	Tendinite	n°57
Dos	Sciatique par hernie discale	n°97 et n°98
	Radiculalgie crurale par hernie discale	n°97 et n°98
Épaule	Épaule enraidie	n°57
	Épaule douloureuse	n°57
	Tendinopathie de la coiffe des rotateurs	n°57
	Rupture partielle de la coiffe des rotateurs	n°57
Genou	Compression du nerf sciatique poplité externe	n°57
	Lésion chronique du ménisque	n°79
	Hygroma aigu du genou	n°57
	Tendinite sous-quadricipitale ou rotulienne	n°57
	Tendinite de la patte d'oie	n°57

TABLE 1.1 – La localisation articulaire des syndromes de TMS et leur(s) tableau(x) des maladies professionnelles de référence [Ministère de la santé, 2020].

l'imposition d'une charge de travail qu'elle ne peut tolérer sans conséquence dangereuse. Les TMS évoluent au cours du temps : il s'agit d'un processus qui se développe progressivement avec la répétition de la sur-utilisation et une récupération insuffisante [Simoneau et al., 2012].

Les TMS-MS résultent généralement d'un déséquilibre entre les sollicitations biomécaniques et les capacités fonctionnelles de l'opérateur. Ces capacités dépendent notamment de l'âge, du sexe, de l'état physiologique et psychologique ainsi que des antécédents personnels. Lorsque les sollicitations sont inférieures aux capacités fonctionnelles, la probabilité de survenue d'un TMS-MS est faible et correspond au niveau de risque minimal [INRS, 2011a].

Les facteurs biomécaniques

Il n'existe pas de TMS sans sollicitation biomécanique [Simoneau et al., 2012]. Les facteurs biomécaniques directs sont les principaux facteurs de risque des TMS. Ils concernent

la manutention manuelle, la force et les efforts exercés, les postures adoptées (contraintes cervicales) ainsi que la répétitivité et la cadence (rythme de la machine, de la chaîne, etc.) des gestes effectués, qui entraînent une sollicitation continuelle des mêmes structures anatomiques. Les muscles et les autres tissus mous sont abîmés à cause de ces contraintes. Il faut aussi prendre en compte les facteurs biomécaniques "indirects" ou environnementaux comme les vibrations, l'ambiance thermique, l'ambiance lumineuse et le niveau de bruit.

Les facteurs les plus aggravants sont [Brière et al., 2015] :

— les positions angulaires articulaires extrêmes ;
— les efforts excessifs ;
— le travail en position statique maintenue ;
— les mouvements en force ;
— les postures extrêmes, comme le travail bras au-dessus des épaules ;
— les mouvements de torsion du poignet ;
— les mouvements de torsion du tronc ;
— les mouvement de flexion et d'extension du coude ;
— la répétitivité élevée des gestes ;
— les vibrations transmises par des outils ou des machines ;
— la pression localisée sur les tissus mous .

Toujours d'après [Brière et al., 2015], en 2010, 25% des salariés répètent un même geste ou une série de gestes (contre 17% en 2003 et 20% en 1994), et plus de 8% des salariés le font plus de 20 heures par semaine. Ces gestes répétitifs concernent autant les femmes que les hommes, surtout des ouvrier-e-s non qualifié-e-s. Dans des études menées par l'INRS en milieu industriel, la répétitivité apparaît comme le facteur biomécanique qui présente le plus de poids dans la survenue des TMS du poignet [INRS, 2011a].

Les facteurs organisationnels

Les contraintes organisationnelles concernent les marges de manœuvre disponibles pour le salarié au niveau de l'organisation de son travail, telles que sa charge ou son rythme de travail. Le risque diminue lorsqu'il est possible de faire varier les exigences du travail. Les TMS surviennent dans des environnements contraints lorsque peu de marges de manœuvre (délais, rythme, quantité produite, etc.) sont permises.

Une organisation mal conçue, ou trop rigide, peut engendrer des risques supplémentaires dus à différents facteurs : les consignes contradictoires, la non-compréhension du travail demandé, le manque d'autonomie, le sentiment de ne pas bien faire son ouvrage, la monotonie, l'impossibilité de réguler la tâche, les interruptions fréquentes ou l'absence de pauses.

Les facteurs psychosociaux

La charge de travail, la pression temporelle, les exigences de concentration liées à la tâche, l'auto-contrôle sur le travail, la participation des salariés aux décisions prises sur leur travail, le climat social, le soutien des collègues et de la hiérarchie, un possible avenir professionnel, le sens donné au travail et la reconnaissance constituent des facteurs psychosociaux freinant l'apparition de troubles musculo-squelettiques [INRS, 2015].

Ces facteurs psychosociaux peuvent être sources de stress lorsque le salarié en a une perception négative (charge de travail excessive, avenir professionnel incertain, manque de reconnaissance, etc.). Un nombre croissant de travaux montre les multiples relations entre les TMS et le stress [Aptel and Cnockaert, 2002, Hartzell et al., 2017, Wixted et al., 2018, Buscemi et al., 2019]. Les forces de serrage et d'appui sont accrues, la tension musculaire s'accroît, le temps de récupération s'allonge (l'action des défenses immunitaires et des systèmes de réparation pour lutter contre les TMS peut être limitée par le stress). L'opérateur stressé peut travailler trop vite, trop intensément, trop longtemps, négliger sa posture ou ne pas prendre le temps d'ajuster son poste. Enfin, le stress amplifie la perception de la douleur et rend les salariés plus sensibles aux risques de TMS [Brière et al., 2015].

Les facteurs individuels

L'âge, le genre ou encore l'état de santé sont des caractéristiques individuelles du salarié qui peuvent être sources de risques. En effet, [Brière et al., 2015] a montré que le taux de TMS augmente avec l'âge et que les femmes sont plus sujettes aux TMS.

L'ancienneté peut aussi induire un risque, notamment à travers la durée d'exposition aux risques du salarié : plus longtemps ce dernier a été exposé, plus le risque de développer des TMS est important. Le patrimoine génétique des salariés et leurs antécédents médicaux et traumatiques peuvent aussi être des facteurs favorisant l'apparition de TMS.

Certaines caractéristiques individuelles peuvent être protectrices : connaître son métier en développant des stratégies protectrices limite l'apparition de TMS, tandis que le non-respect du temps d'apprentissage est une source de risques [Malakoff Médéric, 2015].

1.2.5 Conséquences des troubles musculo-squelettiques

Les TMS peuvent avoir des conséquences graves pour le salarié (handicap, etc.), mais ont aussi des conséquences néfastes pour l'entreprise (absentéisme, etc.). Cette partie détaille les conséquences des TMS chez les acteurs concernés.

Conséquences pour la victime

La première des conséquences pour la victime de TMS est la souffrance physique associée à la douleur ressentie par le salarié au niveau de l'articulation touchée.

Pour la victime, un handicap professionnel et personnel peut aussi découler de l'apparition d'un TMS. Cet handicap a de multiples conséquences comme l'incapacité à effectuer des gestes de la vie courante ou des gestes nécessaires à l'exécution de tâches au travail. Ce handicap peut donc mener à une désinsertion professionnelle et une désocialisation.

Enfin, la personne souffrant de TMS est aussi victime d'une souffrance mentale : la difficulté à tenir son poste de travail et à combler les exigences de son employeur ou bien les difficultés à effectuer des gestes de la vie courante peuvent porter atteinte à l'image de soi [Carsat HF, 2020].

Conséquences pour l'entreprise

Les conséquences des TMS pour l'entreprise sont souvent méconnues et sous-estimées [INRS, 2011b]. Elles sont de natures différentes mais ont des coûts importants, divisés dans la littérature en trois groupes [Réseau Anact, 2008] :

— les **coûts directs** : ils sont directement imputables aux TMS et à leur gestion. Ce sont les coûts les plus simples à évaluer pour l'entreprise. Ils sont composés des cotisations versées à l'assurance maladie, des indemnités pour les salariés malades (absences ou soins), des frais associés à l'aménagement des postes de travail (recherche et aménagement) et du temps passé à gérer les dossiers des salariés souffrant de TMS. Ils varient en général de 100€ à 500€ par an et par salarié (victime ou non) [Réseau Anact, 2008] ;

— les **coûts de régulation** : ils sont attribuables aux dysfonctionnements dus aux TMS. Les TMS provoquent de l'absentéisme (au total, près de vingt-deux millions de journées de travail perdues en 2018 en France à cause des TMS ou du mal de dos [CNAM, 2018]). En effet, le salarié souffrant de TMS doit souvent cesser de travailler pendant une période plus ou moins longue (en moyenne deux cent vingt jours pour un TMS à l'épaule ou cent cinquante et un jours pour un syndrome du canal carpien [Carsat PL, 2020]), afin d'interrompre l'exposition aux risques. Cet absentéisme a des conséquences sur l'entreprise, car il provoque souvent une désorganisation du travail et une surcharge de travail pour les employés encore en poste, ainsi que de possibles difficultés de remplacement du personnel arrêté [INRS, 2011b]. Il peut aussi y avoir une perte de productivité si il y a une perte de compétence à cause de l'absence du salarié. Il faut noter que la désorganisation et la surcharge de travail peuvent, à terme, provoquer des TMS chez d'autres salariés. Au coût dû à l'absentéisme du salarié peuvent s'ajouter des coûts indirects dus à une dégradation de l'ambiance de travail. Cette régulation coûte en général entre deux et sept fois le

montant des coûts directs [INRS, 2011a, Réseau Anact, 2008] ;

— les **coûts stratégiques** : ils correspondent à l'impact des TMS sur la capacité de développement de l'entreprise. En effet, les TMS ont un impact sur la capacité de l'entreprise à mobiliser ses ressources pour gagner ou maintenir une position concurrentielle [Réseau Anact, 2008]. Les coûts stratégiques renseignent sur les principales limites que posent les TMS aux capacités de l'entreprise. Les limites posées par les TMS peuvent être sociales (tensions dans l'entreprise), productives (augmentation du temps de travail ou baisse de la cadence), économiques (effet sur les prix associé au coût du TMS) ou éthiques (dégradation de l'image de l'entreprise). Ce sont aussi les coûts liés à la baisse de qualité induite par l'absence du titulaire du poste (remplacé par des intérimaires, en général, moins qualifiés).

Lorsque les TMS apparaissent, c'est l'entreprise qui est malade et qui perd de l'argent [INRS, 2011b]. L'ensemble de ces coûts peut remettre en cause la pérennité de l'entreprise.

1.2.6 Prévention des troubles musculo-squelettiques

La prévention des TMS dans l'entreprise s'organise sur la base du dialogue entre l'employeur et les salariés ou leurs instances représentatives : comité d'hygiène, de sécurité et des conditions de travail (CHSCT) ou délégués [CNAM, 2009].

La prévention des TMS se fait en trois temps [CNAM, 2009, INRS, 2015] :

— elle commence par un dépistage des situations de travail à risque souvent effectué par des ergonomes. Ceux-ci mettent en place des entetiens avec les salariés et observent leur travail afin d'analyser les situations de travail. Ce dépistage permet d'identifier et de comprendre les situations à risque ;

— elle continue par une modification des situations de travail pour réduire voire supprimer les facteurs de risque identifiés lors de l'étude menée précedemment. Une période d'information et de formation des salariés fait aussi partie de cette étape ;

— elle se termine par une phase d'évaluation de l'intervention afin de vérifier que les mesures mises en place permettent effectivement de réduire les situations à risque.

La prévention des TMS s'inscrit dans la durée. Des actions modestes sont utiles pour mettre en place une première démarche, étape nécessaire à l'appropriation par les acteurs de l'entreprise afin de pouvoir ensuite élargir la réflexion à l'organisation du travail et à l'environnement de travail.

1.3 La norme volontaire *NF X35—702*

Déjà évoquée auparavant, la norme volontaire *NF X35—702* [AFNOR, 2015], entrée en vigueur le 6 juin 2015, est une norme française homologuée définissant les exigences

et préconisations à prendre en compte lors de la conception des cabines de tri manuel des déchets recyclables. Cette norme a été élaborée par une commission comprenant à la fois des ergonomes, des exploitants de centre de tri, et des membres de bureaux d'étude de conception de ces mêmes centres. Elle s'appuie sur des principes ergonomiques pour présenter les caractéristiques nécessaires à la prévention de la santé et à la sécurité des agents de tri. Elle fournit ainsi une trame pour l'appréciation des conditions de travail des valoristes, en donnant une analyse des facteurs de risque pour la sécurité et la santé des agents de tri. Finalement, elle expose une approche pour le calcul du nombre d'agents et de postes de tri nécessaires en cabine en fonction d'une estimation du nombre d'actions de tri à effectuer.

Cette norme nous a servi à l'élaboration de certains indicateurs pour la mesure de la pénibilité de l'activité de l'agent de tri. En effet, cette norme évoque plus particulièrement les risques d'apparition à moyen et long terme de troubles musculo-squelettiques (TMS) concernant les membres supérieurs suite à l'exposition prolongée à certaines contraintes telles que les manutentions manuelles, les postures contraignantes, les manipulations répétitives et la charge mentale élevée.

La norme *NF X35—702* [AFNOR, 2015] impose une série de contraintes techniques et donne des indicateurs permettant une évaluation de la pénibilité de l'activité du valoriste. Ces contraintes et indicateurs sont présentés dans cette partie.

1.3.1 Contraintes techniques introduites par la norme *NF X35—702*

Pour améliorer les conditions de travail des agents de tri, la norme *NF X35—702* impose des contraintes techniques sur différents paramètres de la cabine de tri qui peuvent influencer notre développement. Ces différentes contraintes sont présentées dans cette partie.

Dimensions de la table de tri et de sa bande

Les dimensions de la table de tri sont données par la norme *NF X35—702* et illustrées en Figure 1.7. La table de tri et sa bande doivent respecter les règles suivantes :

- la hauteur de la table de tri doit être de 1 000 mm à partir du niveau du sol de la cabine.
- pour une activité de tri unilatéral, la largeur de bande recommandée est 500 mm, et ne doit pas dépasser 600 mm.
- pour une activité de tri bilatéral, la largeur de bande recommandée est 1 000 mm, et ne doit pas dépasser 1 200 mm. Une tolérance est cependant accordée aux tables réservées aux matériaux de grande taille tels que les films industriels ou les gros cartons.

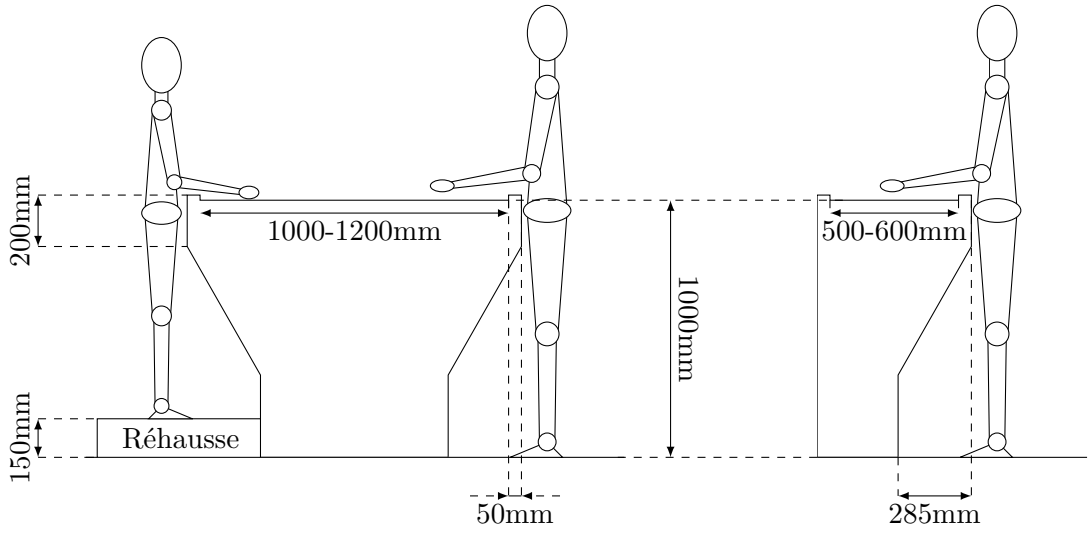


FIGURE 1.7 – Les dimensions de la table de tri selon la norme *NF X35—702* pour une table de tri bilatéral (à gauche) et pour une table de tri unilatéral (à droite).

— les rives de la table ont une largeur maximale recommandée de 50 mm, et une hauteur qui doit être la plus faible possible tout en empêchant le débordement des matériaux hors de la ligne de tri.

Cette table va donc obstruer la vue des membres inférieurs des valoristes si le capteur est situé de l'autre côté de la table de tri. La partie supérieure du corps des valoristes (torse, bras, tête) devrait cependant rester visible, même si ceux-ci sont de petite taille, grâce à la présence d'une réhausse, d'une hauteur maximale de 150 mm au niveau du poste de tri pour faciliter leur travail et du positionnement du capteur à une hauteur raisonnable, c'est-à-dire au niveau de la table de tri ou bien au-dessus.

Position des exutoires

Les exutoires sont situés en face, sur le côté ou derrière les valoristes. La norme *NF X35—702* recommande que les agents de tri évacuent le matériau majoritaire à trier dans un exutoire situé en amont par rapport au sens de défilement de la table de tri, ce qui signifie que le membre situé au niveau de cet exutoire sera probablement plus sollicité que l'autre. De plus, la présence d'exutoires situés en face du trieur ou de l'autre côté de la bande permet au valoriste de lancer des objets. Un exemple de positionnement des exutoires pour les différents postes est donné en Figure 1.3.

Nettoyage et propreté de la cabine

Comme préconisé par la norme *NF X35—702* [AFNOR, 2015] et facilement expliquable par le fonctionnement général du centre de tri, la cabine de tri est conçue pour pouvoir être fréquemment nettoyée. Ainsi, afin de réduire la stagnation de poussière et de faciliter

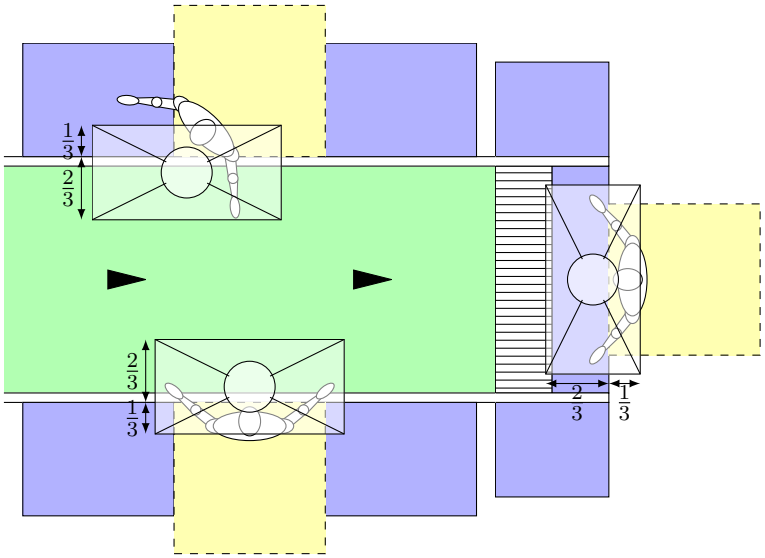


FIGURE 1.8 – Des plénums sont situés au-dessus des postes de travail des opérateurs. Exemple pour les trois différents types de postes, le long d'une table de tri bilatéral.

son entretien, les cloisons de la cabine de tri sont affleurantes et doivent pouvoir faire l'objet d'un nettoyage humide. De même, les luminaires, les grilles de ventilation et les autres équipements du plafond doivent être accessibles pour leur nettoyage et il doit être possible de réaliser un contrôle et une maintenance pratique de la totalité des équipements de ventilation.

La présence de notre dispositif ne doit donc pas gêner l'accès aux équipements présents au niveau du faux-plafond de la cabine de tri, et il doit pouvoir faire l'objet d'un nettoyage humide s'il est placé au niveau d'une cloison de la cabine.

Présence de ventilation au plafond et de plafonniers

La présence de bouches de ventilation, aussi appelées plénums, au-dessus des valoristes (voir Figure 1.8) imposées par la norme *NF X35—702* [AFNOR, 2015] ainsi que de luminaires au plafond contraignent la position des capteurs. Les ventilations empêchent le positionnement d'un capteur directement au-dessus du trieur alors que les plafonniers restreignent l'espace disponible où les capteurs pourraient être fixés, si on les accroche au plafond.

Circulations internes

Il est courant que du personnel traverse la cabine de tri par des allées de circulation, par exemple pour se rendre dans une autre cabine de tri ou pour aller vérifier le fonctionnement d'une machine en amont ou en aval de la cabine de tri.

— Atteindre
— Déchirer
— Déplacer
— Glisser
— Lancer
— Pousser
— Saisir
— Secouer
— Tirer

TABLE 1.2 – Liste non exhaustive des actions techniques courantes réalisées par les valoristes d'après la norme *NF X35—702*.

D'après la norme *NF X35—702*, les allées de circulation internes à la cabine doivent être de préférence au même niveau, permettre le croisement de deux personnes et faciliter l'évacuation d'urgence. La largeur de passage, libre de tout obstacle, ne doit pas être inférieure à 1 000 mm.

Ainsi, il est possible que des personnes qui ne sont pas des valoristes ou qui ne sont pas en train de trier entrent dans le champ de vision des capteurs. Ceci implique donc qu'il doit être possible de distinguer le trieur actif à son poste des personnes circulant dans les allées.

De plus, les capteurs ne pourront pas être placés dans ces allées, comme ils ne doivent pas être des obstacles pour les opérateurs qui y circulent.

1.3.2 Indicateurs pour l'évaluation et la mesure de la pénibilité proposés par la norme *NF X35—702*

Des indicateurs pour l'évaluation de la pénibilité de l'activité de l'agent de tri sont un second type d'éléments présentés par la norme *NF X35—702*. Ces indicateurs sont décrits dans cette partie.

Actions techniques à détecter et/ou reconnaître

De nombreuses actions sont couramment effectuées par les trieurs lorsqu'ils trient les matériaux qui passent par la chaîne de tri. Une partie de ces actions est répertoriée par la norme *NF X35—702* et sont rappelées en Tableau 1.2. Cette liste n'est cependant pas exhaustive : les conditions de fonctionnement de la cabine de tri et de la stratégie de tri de l'opérateur ou plus généralement de l'équipe de valoristes, peuvent amener les opérateurs à effectuer d'autres actions, comme, par exemple, retenir la matière ou bien l'accumuler dans une main avant de la déposer dans un exutoire.

Un centre de tri comprenant généralement plusieurs cabines de tri contenant plusieurs tables de tri sur lesquelles travaillent souvent plusieurs trieurs, la vitesse et le nombre d'exécution des actions de tri varient selon la table et le poste occupé par le valoriste. Comme écrit précédemment, il faut aussi distinguer les actions de tri des actions parasites.

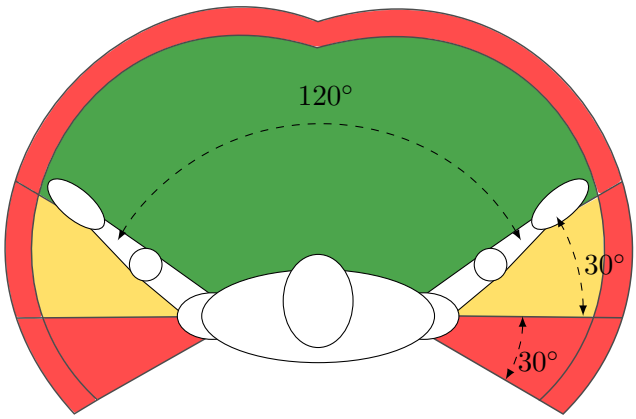


FIGURE 1.9 – Les différentes zones autour d'un valoriste, telles que définies par la norme *NF X35—702* : zone verte (risque réduit), zone jaune (risque accru), zone rouge (risque critique), hors zone (risque délétère).

Zones à repérer

La norme *NF X35—702* définit quatre zones autour du valoriste (voir Figure 1.9). Ces zones d'évolution des membres supérieurs de l'agent de tri permettent d'estimer les risques biomécaniques encourus par le valoriste. En zone verte, le risque est réduit, en zone jaune, il est accru, en zone rouge, il est critique, hors-zone, le risque est délétère. La détection correcte de ces zones est primordiale pour la comptabilisation des actions techniques des trieurs.

Comptabilisation des actions techniques

La norme *NF X35—702* décrit une manière de mesurer la pénibilité par la mesure d'un nombre d'actions effectuées pendant une certaine durée. La norme *NF X35—702* définit l'action technique (abrégée AT) comme une unité de comptabilisation de l'activité d'un valoriste ainsi qu'une méthode permettant de calculer la quantité d'activité d'un trieur grâce au comptage des actions techniques. La méthode de comptabilisation proposée est donnée dans le Tableau 1.3. Le calcul des actions techniques effectuées par un trieur est fonction de l'action réalisée et de la zone d'exécution de celle-ci (voir Figure 1.9).

On observe que le coût d'une action augmente logiquement avec la difficulté de la zone dans laquelle elle est effectuée. On remarque cependant que la comptabilisation est la même lorsque les gestes sont effectués en zone jaune ou rouge, ce qui pourrait permettre une simplification dans la séparation des zones, en ne voyant plus celles-ci que comme une seule et unique zone.

Zone	AT par action	Détails
Zone verte	1 AT	
Zone jaune	2 ATs	1 AT atteindre + 1 AT prendre/déposer
Zone rouge	2 ATs	1 AT atteindre + 1 AT prendre/déposer
Hors zone	4 ATs	2 AT atteindre + 2 AT prendre/déposer
Lancer	2 ATs	

TABLE 1.3 – Méthode de comptabilisation des actions techniques (AT) proposée par la norme *NF X35—702*.

1.4 Contraintes du projet

Cette section présente les contraintes inhérentes au projet, qu'elles soient induites par l'implantation en environnement industriel, la technologie et les contraintes de positionnement des capteurs utilisés ou imposées par l'entreprise.

1.4.1 Contraintes introduites par l'environnement industriel

Si la norme *NF X35—702* impose une série de contraintes techniques qui attraient à différents paramètres de la cabine de tri, le contexte industriel dans lequel s'inscrit cette thèse vient lui aussi avec son ensemble de contraintes. En effet, l'environnement industriel, où des évenements non-contrôlés peuvent se produire, ajoute une variabilité décrite ici comme une série de contraintes dont il faut tenir compte. Ces contraintes sont exposées ici.

Variabilité de la position et des dimensions de la chaîne de tri

Les différentes possibilités au niveau de la position dans la cabine de tri (contre une cloison, entre deux allées) ou des dimensions de la table en fonction d'un tri unilatéral (simple tapis de 500 à 600 mm, double-tapis de 500 à 600 mm séparés par un exutoire central) ou bilatéral (tapis de 1 000 à 1 200 mm) ainsi que la taille variable de la cabine de tri et les différentes hauteurs de plafond possibles, sont des contraintes physiques à prendre en compte lors du positionnement du capteur et de l'acquisition des données.

Des exemples de positionnements et de dimensions de la table de tri sont illustrés en Figure 1.10.

Perturbation du signal par les flux d'air et les sources lumineuses parasites

Les ventilations précédemment citées, les autres sources de lumière, qu'elles soient naturelles à cause de la vue sur l'extérieur ou artificielles et dues aux éclairages ambiants, et la présence d'objets réfléchissants (par exemple les bandes sur les gilets des valoristes, voir Figure 1.11) peuvent aussi perturber le signal. En effet, il est probable que ces compo-

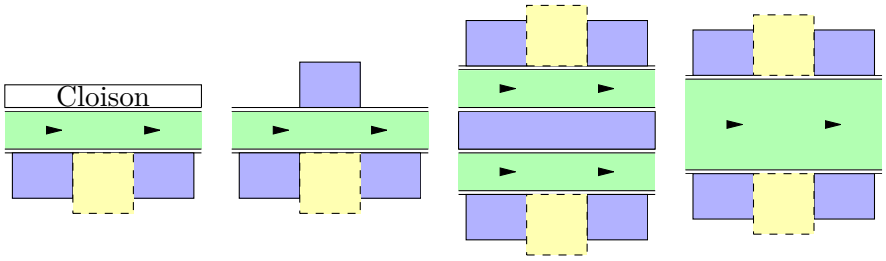


FIGURE 1.10 – Des exemples de positionnements possibles et de dimensions possibles des tables de tri.

FIGURE 1.11 – Des bandes réfléchissantes sur les équipements de sécurité peuvent perturber le signal.

sants faussent les données reçues par le récepteur. La présence d'équipements réfléchissants composant la chaîne de tri n'est pas à négliger (plastique brillant par exemple).

Variabilité de la position des valoristes

La position des trieurs est aussi une source de contraintes. Du fait qu'il existe de multiples façons de positionner les trieurs sur la chaîne de tri (voir Figure 1.12), il est très difficile de trouver une position pour le capteur qui puisse être commune à tous les postes.

Les valoristes sont généralement positionnés face-à-face le long d'une chaîne de tri, mais ce n'est pas le cas tout le temps : il arrive que deux valoristes soient positionnés côte-à-côte ou face à un mur, et cette disposition est tout de même délicate à gérer puisqu'elle s'éloigne des conditions de mise en œuvre usuelles des capteurs qui font habituellement face aux utilisateurs. Il faut, par ailleurs, considérer les contraintes dues à la morphologie des valoristes et du poste de tri.

Enfin, afin d'anticiper la reconnaissance et le tri des déchets pour les valoristes, le premier poste de travail doit se trouver à une distance minimum de 2 000 mm de l'entrée

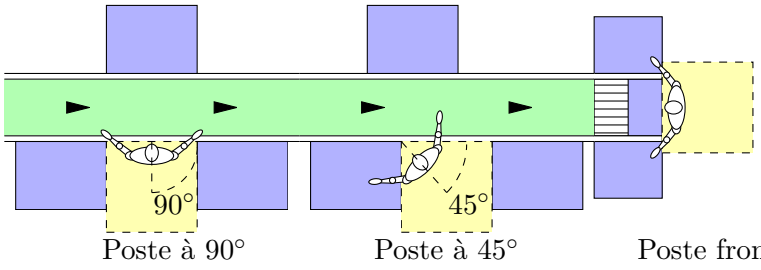


FIGURE 1.12 – Positionnements possibles des trieurs sur la chaine de tri, le long d'une table de tri unilatéral. Sur le poste à 90°, l'opérateur se place perpendiculairement au flux de matière, tandis qu'il se tourne vers l'amont sur le poste à 45°. Enfin, sur le poste frontal, le trieur fait face à la table de tri.

des déchets en cabine de tri, tandis que pour effectuer un éventuel rattrapage de déchets, il convient que le dernier poste de travail dispose d'une longueur de la table de tri disponible de 900 mm en aval. Cependant, ces distances ne sont pas toujours respectées pour les centres ayant été conçus avant l'apparition de la norme *NF X35—702*.

1.4.2 Contraintes sur les capteurs imposées par l'entreprise

L'entreprise a bien évidemment voix au chapitre lors de la sélection des capteurs à utiliser. Ainsi, les contraintes qu'elle a imposées pour le choix du capteur sont présentées ici.

Coût du capteur

La volonté de l'entreprise est que le capteur doit être relativement peu onéreux, afin qu'il puisse aussi bien être installé sur les installations présentes et futures de la société Ebhys, que sur les installations de ses concurrents. Il est important que le capteur soit bon marché pour pouvoir équiper les cabines de tri à moindre coût, afin que les propositions de l'entreprise puissent rester concurrentielles, et que le coût soit rapidement absorbé par des économies réalisées par le client au niveau de la gestion des troubles musculo-squelettiques.

Installation du capteur

Si la société ne souhaite dans un premier temps qu'équiper les nouveaux centres de tri qu'elle conçoit, la volonté de pouvoir installer le dispositif sur des chaînes de tri déjà en fonctionnement a été évoquée. L'entreprise voudrait par ailleurs être en mesure d'équiper des centres de tri conçus par ses concurrents.

De plus, la norme *NF X35—702* recommande de prévoir une cabine de tri flexible et évolutive. Cette recommandation se traduit par la présence d'un espace suffisant le long de la ligne pour accueillir des postes de tri supplémentaires mais aussi par l'extension

possible de la cabine de tri. Cela signifie, par ailleurs, qu'il est envisageable que tous les postes de tri ne soient pas occupés, et donc que certains postes ne soient dans un premier temps pas équipés.

La volonté d'équiper des centres de tri déjà en fonctionnement ainsi que ceux des concurrents, ajoutée à une évolutivité recommandée par la norme *NF X35—702* imposent que le capteur et le dispositif de mesure soient faciles à installer et à mettre en marche.

Positionnement des capteurs

Afin d'éviter tout accident ou acte de malveillance de la part du personnel en cabine vis-à-vis du dispositif et plus spécifiquement des capteurs, la société Ebhys souhaite que les capteurs soient positionnés de manière à être hors de portée des opérateurs. Il est donc nécessaire de placer les capteurs en hauteur avec une vue plongeante.

Type de capteur

Il a été décidé en amont que les capteurs doivent être principalement attachés aux dispositifs intégrables par l'entreprise Ebhys sur les unités vendues à ses clients. Il n'est donc pas souhaité que des capteurs soient intégrés aux habits de travail (comme par exemple des gants) malgré les simplifications potentielles que cela pourrait provoquer.

Automatisme du dispositif

La société Ebhys n'étant pas une entreprise de services du numérique, ses employés ne sont pas familiers avec le développement informatique. Il en va de même pour les exploitants de centre de tri. Ainsi, il est primordial que le dispositif développé au cours de cette thèse fonctionne de la manière la plus automatique possible. La mise en place du système peut toutefois nécessiter une opération de calibrage, qui peut être réalisée lors de la mise en place du site.

1.4.3 Contraintes liées aux capteurs utilisés

Les contraintes imposées par l'entreprise sont toutes des contraintes techniques vis-à-vis du capteur qui permet d'acquérir les données afin de réaliser la mesure de la pénibilité. À ces contraintes s'ajoutent des contraintes propres au capteur, qui doit être utilisables dans les conditions difficiles de l'environnement industriel.

Ces contraintes intrinsèques sont présentées dans cette partie.

Acceptabilité du capteur

La capteur utilisé doit être acceptable pour les trieurs. En effet, ceux-ci ne doivent pas associer les capteurs présents dans la cabine de tri à un mécanisme de surveillance. Des

caméras optiques à base de technologies RGB ne peuvent par exemple pas être utilisées car elles risquent d'être vues comme un moyen de surveillance par les employés.

De plus, selon la Commission Nationale de l'Informatique et des Libertés (CNIL) au travers de la loi "Informatique et Libertés", et selon l'Union Européenne suite à l'adoption du Réglement Général sur la Protection des Données (RGPD), il est interdit de filmer les employés sur leur poste de travail, sauf sous certaines circonstances particulières, car les employés ont droit au respect de leur vie privée. Enfin, seules les personnes habilitées par l'employeur peuvent avoir accès aux images enregistrées, dont la durée de conservation doit être définie.

Ainsi, les personnes filmées doivent être informées de la présence du dispositif grâce à un panneau à afficher dans les locaux. Ce panneau doit au moins mentionner :

— l'existence du dispositif ;
— le nom du responsable ;
— la raison d'être du dispositif ;
— la durée de conservation des images ;
— la possibilité d'adresser une réclamation à la CNIL ;
— la procédure à suivre pour accéder aux images les concernant.

Enfin, certaines formalités sont à accomplir en fonction que le lieu filmé soit ouvert ou non au public [UE, 2016, CNIL, 2019].

Interférence inter-capteurs

L'utilisation de multiples capteurs actifs dans un même environnement peut poser problème. En effet, un risque d'interférence apparaît lorsque l'on utilise plusieurs capteurs au même endroit : les capteurs actifs sont en général composés d'un émetteur et d'un récepteur, l'émetteur émettant un signal qui est ensuite capté par le récepteur afin d'en extraire des informations. Ainsi, deux capteurs émettant des signaux au même endroit dans le même domaine de fréquence peuvent provoquer des interférences et les signaux émis peuvent ne pas être correctement captés par les recepteurs. Ces interférences entraînent donc, dans le meilleur des cas, une perte d'information sous forme de bruit et, dans le pire des cas, la réception de signaux aberrants qui faussent l'acquisition (voir un exemple illustratif en Figure 1.13).

Robustesse du capteur

Le capteur doit être assez robuste et stable pour pouvoir résister aux conditions normales de fonctionnement de la cabine de tri. En particulier, l'influence des vibrations et des flux d'air engendrés par le fonctionnement des autres équipements du centre de tri et de la cabine de tri sur les données acquises doit être limitée au maximum.

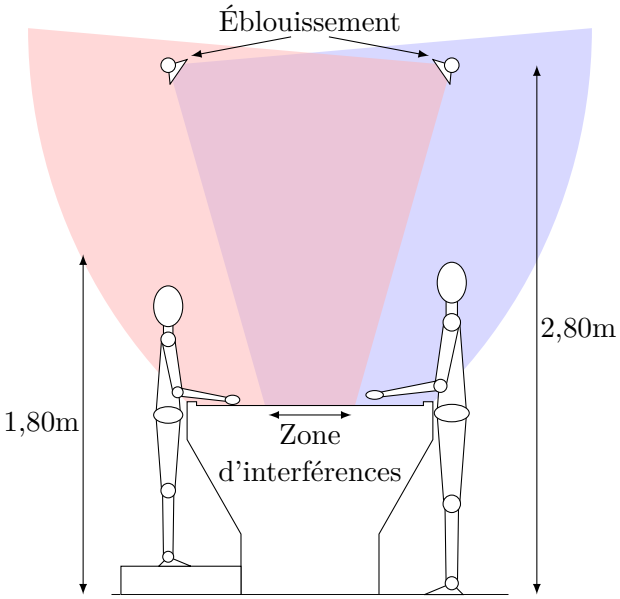


FIGURE 1.13 – Du fait de leur fréquence d'émission et de leur angle de vue, l'utilisation en parallèle de certains capteurs peut provoquer des interférences inter-capteurs, voire un éblouissement total.

1.5 Aspect matériel

Comme écrit précédemment, il a été décidé que les capteurs devraient être principalement attachés aux dispositifs intégrables par la société Ebhys sur les unités vendues à ses clients. En effet, pour favoriser l'acceptabilité du capteur, il n'est pas souhaité d'utiliser des capteurs intrusifs comme des vestes ou des gants, même si cela pourrait éventuellement faciliter les mesures d'activité.

De plus, la présence de sources lumineuses parasites peut provoquer un éblouissement des capteurs, par exemple si ils sont positionnés à contre-jour. La volonté que les capteurs soient acceptés par les opérateurs, et qu'ils s'inscrivent dans le cadre légal de la loi "Informatique et Libertés" et du Réglement Général sur la Protection des Données, ne va pas non plus dans le sens des caméras couleurs, qui pourraient être vues comme des instruments de surveillance et ne semblent pas utilisables vis à vis de la loi.

C'est pour ces raisons que les capteurs de profondeur nous semblent les plus adaptés à notre situation. L'étude bibliographique suivante se concentre donc sur ces capteurs externes actifs fixes qui apparaissent plus aisément acceptables et adaptés que les autres types de capteurs.

Une image de profondeur, aussi appelée carte de profondeur (en anglais *depth map*, *depth image* ou *depth frame*) est une image qui contient l'information relative à la distance de la surface des objets d'une scène par rapport à un point de vue. Une image de profondeur est donc une image dont la valeur de chaque pixel représente sa distance par rapport

au capteur qui a permis son acquisition. Cette image a l'intérêt de pouvoir facilement être transformée en un nuage de points, si l'on connait les paramètres intrinsèques du capteur (tels que sa distance focale, les coordonnées de son centre optique et son facteur d'agrandissement). Une image de profondeur ne représente cependant pas une vision 3D complète de la scène, car elle ne peut observer des objets qui seraient occultés par d'autres éléments. On parle donc parfois de vision en 2,5D.

Cette section présente d'abord les différentes technologies utilisées par les capteurs de profondeur pour l'acquisition d'images de profondeur, puis décrit séparément chacun des capteurs étudiés au cours de cette thèse.

1.5.1 Les technologies utilisées par les capteurs de profondeur

Il existe plusieurs technologies permettant d'acquérir des données sur la profondeur de la scène observée. Les trois principales sont les caméras stéréoscopiques, les capteurs à lumière structurée et les capteurs à temps de vol [Chen et al., 2013, Aggarwal and Xia, 2014]. Ces capteurs sont généralement qualifiés de capteurs 2,5D, car il n'est pas possible d'observer les objets situés sur la scène mais cachés par d'autres.

Caméras stéréoscopiques

Les caméras stéréoscopiques sont des caméras composées de plus de deux lentilles avec un capteur indépendant pour chaque lentille (voir Figure 1.14). Le but est de simuler la vision humaine, et ainsi d'avoir la capacité de capturer des images tri-dimensionnelles. Ce sont des capteurs passifs, car ils utilisent la lumière émise par une autre source (par exemple le soleil ou des luminaires) et réfléchie par la scène, ils n'émettent pas de lumière eux-mêmes.

L'acquisition de données de profondeur grâce aux caméras stéréoscopiques est un important domaine de recherche et les caméras stéréoscopiques sont principalement conçues dans ce but là [Aggarwal and Xia, 2014].

Cependant, à cause de la complexité du calcul de la géométrie, la reconstruction de cartes de profondeur à partir de caméras stéréoscopiques est encore un problème difficile, en particulier à cause des calculs de triangulation [Hartley and Zisserman, 2004] et de la difficulté à repérer les pixels qui sont la représentation des mêmes points, aussi appelé *problème de correspondance* [Hussmann et al., 2008] : il est par exemple possible que des points visibles sur une des images soient cachés sur une autre.

L'influence de l'éclairage ambiant sur le capteur est très forte, en particulier au niveau des calculs de triangulation, car elle peut changer la couleur des points en fonction de la position du capteur. L'arrière plan de la scène peut aussi être un problème, surtout si il change beaucoup (il faut alors le recalculer à chaque fois). Pour finir, des zones d'ombres, concernant les zones à l'intérieur du champ de vision mais qui ne sont pas vues par assez

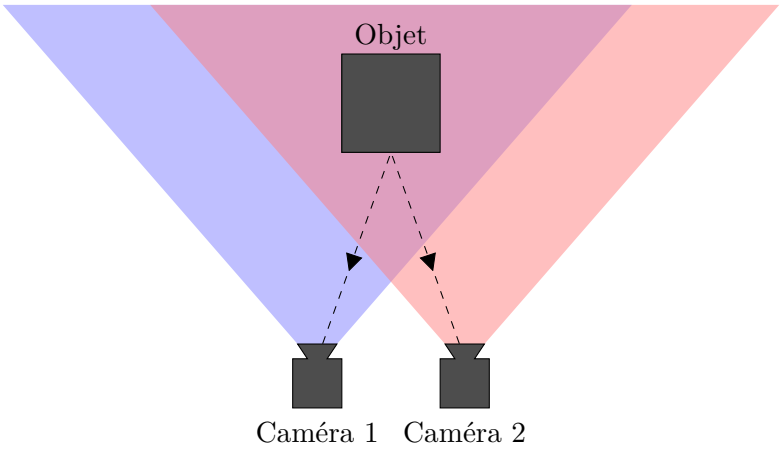


FIGURE 1.14 – Le principe de fonctionnement des caméras stéréoscopiques. Deux caméras filment la même scène depuis deux points de vue différents et une triangulation permet de reconstruire un modèle 3D de la partie commune de la scène.

de capteurs pour être triangulées, sont présentes et peuvent handicaper la détection des mouvements ou des personnes.

Enfin, il se pose un problème d'étalonnage entre les différents capteurs utilisés par la caméra. Enfin, le champ de vision est en général limité car il est égal au champ de vision partagé par tous les capteurs. Ainsi, il est impossible d'utiliser ces capteurs avec un objectif de temps réel pour le moment [Chen et al., 2013], ceux-ci ne sont donc pas adaptés à notre projet.

Capteurs à lumière structurée

Un capteur de profondeur à lumière structurée est un capteur à triangulation active. Il est constitué d'au moins deux composants : un émetteur (parfois appelé projecteur) et un récepteur. L'émetteur émet de la lumière avec une structure prédéfinie qui est déformée par les objets présents sur la scène avant d'être reçue par le récepteur, ce qui permet de recréer la profondeur des objets de la scène (voir Figure 1.15) par triangulation entre le motif émis et le motif reçu [Gühring, 2000]. Il est important de remarquer que les motifs projetés peuvent être de natures très différentes (lignes, formes, nuage de points…).

La lumière émise par l'émetteur peut être dans le domaine de l'infrarouge ou bien dans le domaine visible. Pour ce projet, nous ne considérons que les capteurs à lumière structurée fonctionnant dans le domaine infrarouge, afin que la lumière émise par les capteurs ne gêne pas les valoristes, mais aussi car elle n'est sensible ni à la couleur de la scène, ni à l'éclairage ambiant.

La portée de ces capteurs est en général de quelques mètres, une distance plus grande atténuant trop l'intensité du motif émis et entraînant donc une perte de l'information. La

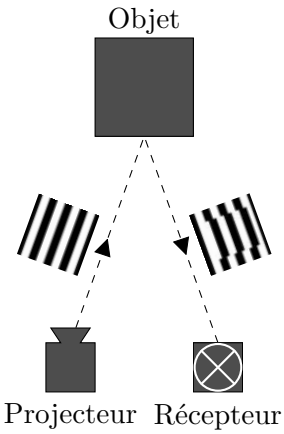


FIGURE 1.15 – Le principe de fonctionnement des capteurs à lumière structurée. Un projecteur émet un signal lumineux structuré qui est déformé par les objets de la scène puis mesuré par le récepteur. La déformation de ce signal par les objets permet de construire l'image de profondeur correspondante.

FIGURE 1.16 – Une acquisition avec un seul capteur à lumière structurée (A), l'interférence créée par l'acquisition avec plusieurs capteurs (B) et l'acquisition avec une légère vibration des capteurs (C). Les capteurs utilisés ici sont des Kinect V1. Image issue de [Maimone and Fuchs, 2012].

résolution des images générées par ces capteurs est correcte, mais les valeurs de profondeur calculées peuvent être imprécises, notamment dans les zones qui ne sont pas illuminées par le motif. Comme pour les caméras stéréoscopiques, le capteur peut présenter des zones d'ombres, dues à la distance entre l'émetteur et le récepteur, et au fait que certaines zones sont vues seulement par l'émetteur ou par le récepteur.

Ce type de capteurs est sujet aux interférences, car plusieurs capteurs placés dans le même environnement émettront plusieurs motifs qui pourraient se superposer. Si il y a une superposition entre les motifs, ceux-ci ne sont plus (ou au mieux mal) reconnus par les capteurs et il y a alors une perte d'information au niveau de la zone de superposition. Ce problème peut néanmoins être surmonté en faisant légèrement vibrer les différents capteurs [Maimone and Fuchs, 2012, Butler et al., 2012] (voir Figure 1.16). Enfin, il n'est pas utilisable en extérieur, à cause des sources de lumière infrarouge comme la lumière du Soleil.

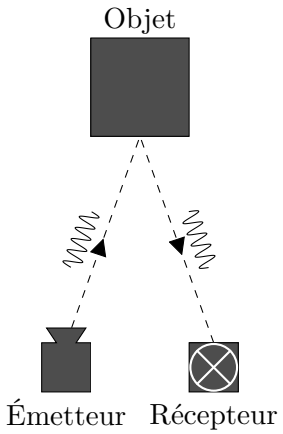


FIGURE 1.17 – Le principe de fonctionnement des capteurs de profondeur à temps de vol. Un projecteur émet un signal lumineux qui est réfléchi par les objets de la scène puis reçu par le récepteur. L'intervalle de temps entre l'émission et la réception de la scène permet de déterminer la distance entre le capteur et les différents objets de la scène et ainsi de construire l'image de profondeur correspondante.

Capteurs à temps de vol

Les capteurs à temps de vol (en anglais *time-of-flight*, *ToF*) se servent de la vitesse de la lumière pour calculer la profondeur de la scène. C'est une méthode active, basée sur l'émission puis la réception d'un signal lumineux (voir Figure 1.17). Il existe deux types de capteurs à temps de vol [Lachat et al., 2015] :

— les capteurs à temps de vol direct : le signal lumineux est envoyé par pulsations (d'où le nom de lumière pulsée) et, connaissant la célérité constante c de la lumière, le calcul de la distance d est direct : $d = \frac{c*\Delta t}{2}$, puisque le signal lumineux fait un aller-retour ;

— les capteurs à temps de vol indirect : le signal lumineux est modulé en amplitude de manière continue, le décalage de la phase est mesuré et la relation liant le décalage de phase et la distance n'est pas directe [Lindner et al., 2010].

Les capteurs de télédétection par laser (en anglais *light detection and ranging*, abrégé LiDAR), ne sont pas décrits ici, car leur prix actuel ne nous permet pas de les utiliser. Cependant, leur méthode de fonctionnement peut être assimilée à celle d'un capteur de profondeur à temps de vol.

Les capteurs à temps de vol direct, bien que simples à comprendre, ont un inconvénient majeur : comme ils sont basés sur la vitesse de la lumière, il est nécessaire qu'ils mesurent très précisément le temps de vol, et donc qu'il soient équipés d'une horloge très précise et parfaitement synchronisée, ce qui augmente grandement leur coût [Lachat et al., 2015].

Ainsi, la très grande majorité des capteurs de profondeur utilisés sont des capteurs à modulation d'amplitude [Lindner et al., 2010]. Ils émettent en général des signaux lumineux

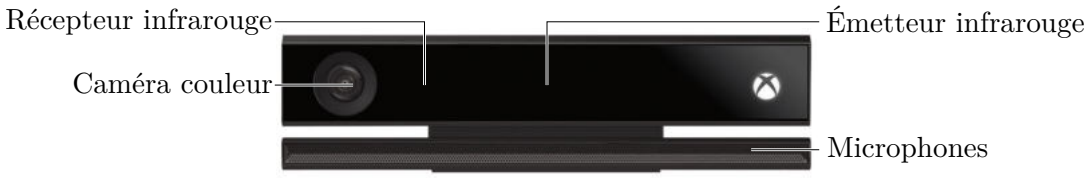


FIGURE 1.18 – La Kinect Version 2 et les différents éléments qui la composent.

dans le domaine de l'infrarouge afin que les signaux ne soient pas visibles par l'homme. Leur portée est proportionnelle à la périodicité du signal émis, la distance maximale étant de $\frac{c}{2*f}$ avec f la fréquence du signal.

Le principal problème de ce type de capteur, s'il était utilisé dans notre projet, est la gestion des interférences. En effet, il est probable que les zones suivies par plusieurs capteurs réfléchissent le signal lumineux émis par un capteur sur le récepteur d'un autre capteur et que les distances calculées soient faussées. De plus, il n'est en général pas utilisable en extérieur, à cause des sources d'infrarouge telles que la lumière du Soleil.

1.5.2 Les différents capteurs de profondeur utilisés

Durant cette thèse, nous nous sommes intéressés à deux capteurs de profondeur, qui ont tous les deux été utilisés durant cette thèse. Ces capteurs de profondeur présentent chacun des caractéristiques différentes qui sont décrites dans cette partie.

Kinect Version 2

La Kinect Version 2 ou V2 est le capteur Kinect développé et commercialisé depuis 2014 par Microsoft pour la console de salon XBOX One, destinée à remplacer la précédente génération avec la Kinect V1 et la XBOX 360. La Kinect V2 est donc une version améliorée de la Kinect V1. C'est un capteur couleur et profondeur qui utilise la technologie du temps de vol (présentée précédemment en Section 1.5.1). Au vu de l'objectif de Microsoft, qui est de développer un capteur grand public dans un but de divertissement et donc de son coût, on peut supposer que la Kinect V2 contient un capteur à temps de vol indirect [Lachat et al., 2015]. C'est un capteur bon marché : son prix de vente est de 150€ auxquels il faut ajouter un adaptateur à 50€ permettant son alimentation et sa connexion à un ordinateur.

La Kinect V2 contient, en plus du capteur de profondeur, une caméra couleur et des microphones (voir Figure 1.18), pour des dimensions de 249×66×67mm^3. Il est envisagé d'utiliser le capteur de profondeur de la Kinect pour l'acquisition de données, tandis que la caméra couleur pourraît être utilisée pendant la phase d'entraînement et de test des algorithmes développés. En revanche, les microphones ne seraient pas utilisés dans le cadre de notre étude.

Le capteur de profondeur a une résolution de 512×424 pixels et peut fournir trente

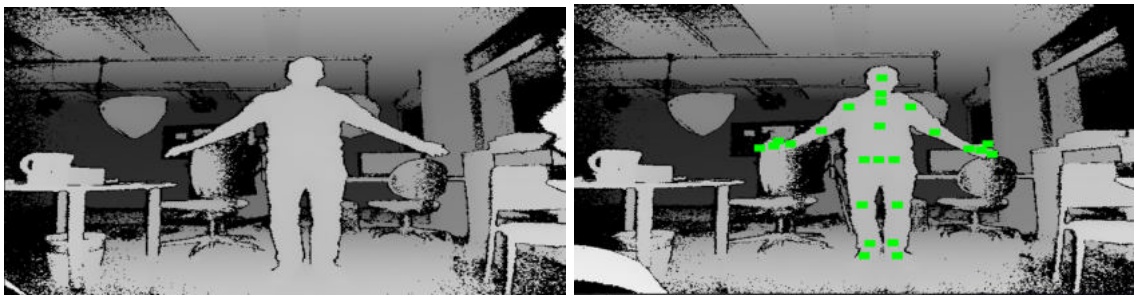


FIGURE 1.19 – Exemple d'image de profondeur acquise avec le capteur de profondeur de la Kinect V2 (à gauche), et une image de profondeur pour laquelle on affiche la position estimée des articulations détectées par le capteur sur l'utilisateur (en vert, à droite).

images par seconde. La profondeur est encodée sur 13 bits pour chaque pixel, ce qui permet d'avoir huit mille cent quatre-vingt-douze valeurs de profondeurs possibles, la profondeur de chaque pixel étant stockée comme un entier non signé et symbolisant la distance de l'objet par rapport au capteur, exprimée en millimètres. Cependant, la portée optimale donnée par le constructeur est de 0,5 à 4,5 mètres. En effet, le pilote officiel masque les distances supérieures à 4,5 mètres, qui sont considérées comme trop lointaines, même si il est théoriquement possible de mesurer des distances jusqu'à 8 mètres. Enfin, l'angle de vue est de 70.6° horizontalement et 60° verticalement.

En plus du capteur, Microsoft fournit également la librairie Kinect for Windows SDK qui contient des fonctions utilisables pour traiter les flux d'informations transmis par la Kinect. Cette bibliothèque a le principal avantage d'être publiée par Microsoft, le créateur de la Kinect, qui connaît donc mieux son fonctionnement que les autres développeurs. Ceux-ci ont du procéder à de la rétro-ingénierie pour pouvoir implémenter des bibliothèques libres comme OpenNI [1] et libfreenect [2], qui manquent donc de certaines fonctionnalités. Un des intérêts du Kinect for Windows SDK est qu'il fournit des indicateurs posturaux des utilisateurs (voir Figure 1.19). Ces indicateurs sont en réalité des estimations de posture données sous la forme de positions 3D de vingt-cinq points représentant les principales articulations et membres, l'ensemble formant un squelette. Cependant, la librairie ne fonctionne correctement que sous Windows, ce qui ajoute un coût supplémentaire lié à l'achat des licences du système d'exploitation lors de la mise en place du dispositif.

De plus, si elle est très fiable lorsqu'on se sert de la Kinect selon ses conditions habituelles d'utilisation (c'est-à-dire en posant le capteur sur une table, face à l'utilisateur), l'estimation de posture se détériore en environnement industriel, où la position du capteur est contrainte et l'environnement n'est pas contrôlé (occultation des membres, perturbations du signal, etc.)

1. https://structure.io/openni
2. https://github.com/OpenKinect/libfreenect2

FIGURE 1.20 – L'Orbbec Astra (en haut) et l'Orbbec Persee (en bas) et les différents éléments qui les composent.

Enfin, il est important de rappeler que le capteur Kinect et son adaptateur ne sont plus fabriqués depuis 2017 et il est donc difficile d'envisager d'équiper des centres de tri avec ce capteur.

Orbbec Astra et Persee

Orbbec est une start-up américaine qui conçoit et commercialise des caméras 3D utilisables pour de la reconnaissance de gestes ou d'activité. La première génération de capteurs commercialisés par Orbbec est l'Orbbec Astra (voir Figure 1.20) et ses différentes déclinaisons.

Orbbec a développé deux capteurs Astra : Astra et Astra Pro. La principale différence entre ces deux caméras est la résolution de la caméra couleur embarquée sur ces capteurs. La version Orbbec Astra contient une caméra à résolution VGA (640×480 pixels) à 30 images par seconde alors que l'Orbbec Astra Pro utilise une caméra avec une résolution de 720p (1280×720 pixels), toujours à 30 images par seconde. De plus, une déclinaison du capteur Astra est dédié aux acquisition de courte portée (Astra S). Tous les capteurs de type Astra ont une dimension de 165×30×40mm^3 et le même poids (0,3kg).

En plus de la caméra couleur, les capteurs Astra contiennent deux microphones, mais surtout un capteur de profondeur utilisant la technologie de la lumière structurée associé à un capteur de proximité pour assurer la sécurité des utilisateurs (à cause des lasers infrarouges utilisés par l'émetteur du capteur de profondeur). Comme pour la Kinect V2, la caméra couleur du Astra pourrait être utilisée pendant la phase d'entraînement et de test des algorithmes implémentés, les microphones ne seraient en revanche pas utilisés pour cette étude.

La portée du capteur de profondeur de l'Astra S est comprise entre 0,35 et 2,5m, alors que la portée de l'Astra et l'Astra Pro est comprise entre 0,6 et 8m, pour une portée opti-

FIGURE 1.21 – Exemple d'image de profondeur acquise avec le capteur de profondeur de l'Orbbec Astra.

male située entre 0,6 et 5,0m. La portée du capteur est donc légèrement plus importante que celle de la Kinect. Le champ de vision des capteurs est de 60° horizontalement et 49,5° verticalement. La profondeur est codée sur 16 bits. La résolution de l'image de profondeur est de 640×480 pixels à 30 images par seconde. Un exemple d'image de profondeur obtenue avec l'Orbbec Astra est donné en Figure 1.21.

L'Orbbec Astra est programmable avec la librairie Orbbec Astra SDK développée par Orbbec, mais aussi OpenNi (sur laquelle s'appuie l'Orbbec Astra SDK). L'Orbbec Astra se connecte à l'ordinateur grâce à une prise USB 2.0 qui alimente le capteur. Il est vendu à un prix unitaire de 150€, indépendamment de la déclinaison du capteur.

Il est important de mentionner que la start-up Orbbec propose aussi le capteur Orbbec Persee. L'Orbbec Persee est un capteur Astra couplé à un ordinateur. Ce capteur est considéré comme la première caméra-ordinateur par l'entreprise. Sa conception a été financée grâce à un financement participatif.

L'Orbbec Persee embarque donc un capteur Astra Pro relié à un ordinateur (voir Figure 1.20). L'ordinateur est composé d'un processeur quadricœur ARM Cortex-A17 32 bits, d'un processeur graphique Mali-T7 600MHz ainsi que de 2Go de RAM. Il possède aussi une prise Ethernet, une carte wi-fi 802.11 et une carte bluetooth 4.0 pour la communication avec d'autres machines et une prise HDMI, un lecteur de carte SD (jusqu'à 32Go) et une prise USB et Micro USB 2.0 pour l'acquisition ou la réception de données. Enfin, les systèmes d'exploitation proposés sont Ubuntu 14.04 ou 16.04 et Android 4.4. Le capteur est alimenté par un adaptateur à 5V 3A. L'Orbbec Persee coûte environ 240€.

En résumé

Pilotée par une problématique industrielle, cette thèse se focalise sur l'analyse de l'activité des opérateurs humains dans un centre de tri dans un but de prévention des troubles musculo-squelettiques (TMS).

Ces pathologies professionnelles, présentées en Section 1.2, touchent généralement les tissus mous et peuvent avoir des conséquences graves, aussi bien pour le salarié que pour l'entreprise. Les causes des TMS sont multiples, cependant, ce travail se concentre sur les facteurs biomécaniques, repérables grâce à des outils de vision : amplitude et répétitivité des actions, postures à risques, etc.

L'environnement industriel du centre de tri (Section 1.1) amène une liste de contraintes fortes, aussi bien sur le type et la qualité des données que l'on peut acquérir, que sur le positionnement des capteurs, leurs dimensions ou encore leur acceptabilité. L'entreprise impose, elle aussi, des contraintes au niveau du choix des capteurs (coût, facilité d'installation, etc.). Toutes ces contraintes sont exposées en Section 1.4. Enfin, des contraintes directement liées à l'objet des travaux sont à prendre en compte : la norme volontaire *NF X35—702* (Section 1.3) définit des actions techniques et des zones de pénibilité ainsi qu'une méthode de comptage.

Les raisons guidant le choix des capteurs nous amènent à nous pencher sur les capteurs de profondeur. Les différentes technologies présentes dans ces capteurs sont décrites en Section 1.5, ainsi que les principaux capteurs utilisés au cours de cette thèse : la Kinect Version 2 et l'Orbbec Astra.

Chapitre 2

Problématique de la thèse

Sommaire

Au vu du contexte présenté dans le chapitre précédent, une problématique claire semble se dégager : l'objectif de cette thèse est d'être en capacité de mesurer des indicateurs pertinents aussi bien au niveau de la norme *NF X35—702* que d'un point de vue ergonomique à l'aide de capteurs de profondeur. Ce chapitre présente le procédé de mise en place des indicateurs de pénibilité, mené en collaboration avec des ergonomes du Centre d'Étude et Recherche Travail Organisation Pouvoir (CERTOP), ainsi que la construction de notre processus de mesure automatique de ces indicateurs.

2.1 Mise en place des indicateurs de pénibilité

Si la norme *NF X35—702* donne déjà une idée des indicateurs à mesurer, une étude ergonomique réalisée en collaboration avec le CERTOP nous a permis d'étoffer et de compléter notre liste d'indicateurs pertinents de la pénibilité au travail dans le cadre de l'analyse de l'activité des agents de tri.

2.1.1 Indicateurs présentés par la norme

La norme *NF X35—702* propose des indicateurs pour la mesure de la pénibilité déjà présentés en Chapitre 1. Nous revenons ici sur ces indicateurs pour les analyser et évaluer leur pertinence.

Actions techniques

La norme *NF X35—702* présente une liste d'actions qui peuvent être réalisées par l'agent de tri 1.3.2. Cependant, comme précisé précédemment, cette liste n'est pas exhaustive car l'agent de tri peut être amené à effectuer d'autres actions en fonction du déroulement de l'activité de tri.

La diversité des actions à effectuer ainsi que la grande variabilité dans la façon de les accomplir, la présence possible de gestes parasites ainsi que la ressemblance entre les exécutions de différentes actions (à titre d'exemple, “Atteindre” et “Déplacer”), rend difficile le travail de reconnaissance des gestes réellement effectués.

On peut aussi noter que certaines actions nécessitent l'usage des deux membres supérieurs (“Déchirer”, etc.), alors que d'autres peuvent occasionner l'utilisation d'un seul ou des deux membres en fonction des dimensions de l'objet trié (“Saisir”, “Lancer”, etc.).

Lors du tri, il se peut aussi que des actions différentes aient lieu simultanément (voire de manière concurrente entre différents trieurs), et il faut être capable de distinguer ces différentes actions et de les comptabiliser correctement.

Enfin, différents acteurs tels que des intervenants de la médecine du travail et des ergonomes ont remis en doute la pertinence de ces indicateurs sur la pénibilité de l'activité.

Ainsi, si la reconnaissance des actions réalisées, nécessitant au préalable une détection, a été envisagée dans un premier temps, cette tâche s'est finalement révélée trop difficile à mettre en place sur les images de profondeur dans un contexte non supervisé. Il a donc été décidé de mettre de côté cet aspect de la pénibilité, qui pourrait cependant faire l'objet de travaux suite à cette thèse.

Zones d'évolution décrites par la norme *NF X35—702*

Également présentées précédemment en Section 1.3.2, des zones d'évolution des membres supérieurs de l'agent de tri sont présentées par la norme *NF X35—702* (voir Figure 1.9). Ces quatre zones permettent d'estimer les risques biomécaniques induits par l'activité de tri du valoriste.

Cependant, la taille réduite des zones d'intérêt rend celles-ci difficiles à discerner et à séparer correctement. La mobilité de ces zones, due aux mouvements de l'opérateur auquel elles se rapportent, rend leur détection et leur séparation complexes.

De plus, elles ne donnent qu'une indication globale de la pénibilité posturale de l'activité du valoriste, et n'indiquent pas quelles sont les articulations réellement sollicitées. Elles ne sont donc qu'un indicateur grossier qui demande à être approfondi.

2.1.2 Indicateurs proposés par l'étude ergonomique

La norme *NF X35—702* propose d'analyser la pénibilité de l'activité de tri par une mesure du nombre d'actions techniques effectuées et de la zone de réalisation de ces actions. La volonté exprimée par les ergonomes lors de l'étude est d'aller au-delà la norme *NF X35—702*. Ainsi, l'étude ergonomique [Dutrieux et al., 2018] a permis de mettre en évidence des indicateurs pertinents et complémentaires qui sont présentés ici.

Mesure des angulations et de la répétitivité

Bien que la norme *NF X35—702* définisse déjà des zones d'évolution des membres supérieurs sur lesquelles on peut se baser pour inférer des intervalles d'angulation pour chacun des membres supérieurs, elle ne donne pas clairement de zones de pénibilité concernant les angulations des articulations.

L'étude ergonomique nous a permis de relever la pertinence de la mesure de ces zones de pénibilité. En effet, les ergonomes ont souligné l'intérêt d'un enregistrement chronologique, appelé "chronique" en ergonomie, permettant de suivre l'évolution des angulations des articulations des membres supérieurs pour mesurer la pénibilité du travail de l'agent de tri. Pour chaque articulation, des zones de risque ont été construites en fonction des angulations. Basées sur des travaux de l'INRS d'après les normes ISO 11226 [ISO, 2000],

ISO 11228—3 [ISO, 2007] et NF EN 1005—4+A1 [AFNOR, 2008], ces zones permettent d'effectuer une classification des risques selon l'amplitude de la mobilisation des articulations. Ces zones sont données en Tableau 2.1. Le code couleur est le même que celui de la norme *NF X35—702* :

- la zone verte représente un risque réduit : elle définit les situations dont les contraintes tendent à limiter la pénibilité pour la plupart des opérateurs ;
- la zone jaune décrit la zone dans laquelle le risque est accru : peu recommandée, elle nécessite de mettre en œuvre des actions de prévention pour réduire le risque ;
- la zone rouge est une zone de risque critique : elle doit être évitée et impose une réduction urgente des contraintes.

De plus, l'étude ergonomique insiste sur la pertinence de mesures temporelles telles que la durée ou la fréquence des expositions à des zones à risque. En effet, une répétition des postures dangereuses ou une exposition prolongée sont des signes importants de la pénibilité de l'activité des agents de tri.

Autres indicateurs ne faisant pas l'objet de nos travaux

L'étude ergonomique a exposé d'autres indicateurs qui ne sont malheureusement pas visibles ni mesurables par notre dispositif. Premièrement, les ajustement posturaux sollicitant la partie inférieure du corps du valoriste ne peuvent pas être vus, car cette partie n'est pas visible à cause de la présence de la table de tri en face, et des exutoires sur les côtés. Deuxièmement, des indicateurs de très faible amplitude tels que les mouvements des mains et des poignets ne sont pas décelables par le capteur de profondeur. Troisièmement, comme il n'est pas possible de distinguer l'état des objets saisis ni la matière qui les compose, il est difficile d'indentifier l'effort de préhension nécessaire à mettre en œuvre pour la saisie des objets. Enfin, le dispositif ne permet pas d'intégrer un ressenti de la pénibilité de la part de l'opérateur.

2.1.3 Indicateurs à mesurer

L'étude ergonomique associée aux limitations techniques du capteur nous a permis d'arrêter une liste d'indicateurs à mesurer pour calculer la pénibilité de l'activité des valoristes.

Ainsi, l'objectif est d'être en capacité de mesurer les indicateurs suivants :

- l'évolution des membres supérieurs dans les zones à risque : cela consiste à mesurer le temps passé dans des zones à risque pour chacune des angulations de l'opérateur ;
- la répétitivité des mobilisations musculaires : elle représente la fréquence à laquelle surviennent les expositions à des zones à risque pour les articulations ;
- le maintien des postures à risque : c'est le nombre d'angulations dans des zones à risque d'une durée supérieure à un sertain seuil ;

Zone verte : risque réduit Zone jaune : risque accru Zone rouge : risque critique

Articulation	Angulation	Zones
Cou	Extension	Inférieure à 20° Comprise entre 20° et 40° Supérieure à 40°
	Flexion	Inférieure à 20° Comprise entre 20° et 40° Supérieure à 40°
	Rotation	Inférieure à 10° Comprise entre 10° et 55° Supérieure à 55°
Buste/Dos	Flexion vers l'avant	Inférieure à 20° Comprise entre 20° et 45° Supérieure à 45°
	Flexion sur le côté	Aucune flexion Inférieure à 10° Supérieure à 10°
	Rotation	Aucune rotation Inférieure à 40° Supérieure à 40°
Coudes	Flexion	Nulle (0°) Comprise entre 0° et 75° Comprise entre 75° et 140° Supérieure à 140°
Épaules	Élévation	Inférieure à 90° Comprise entre 90°et 120° Supérieure à 120°
	Rotation	Inférieure à 80° Comprise entre 80° et 90° Supérieure à 90°

TABLE 2.1 – Les zones de risque pour les angulations de chaque articulation, définies d'après l'étude ergonomique. Les zones sont données par la couleur des angulations.

— les mouvements par zone : cela représente le nombre de mouvements effectués dans chaque zone définie par la norme *NF X35—702* ;
— le nombre d'actions techniques : il consiste à comptabiliser, au sens de la norme *NF X35—702*, le nombre d'actions techniques réalisées par l'opérateur ;

Des indicateurs supplémentaires pourraient être ou seront mesurés par le biais d'autres capteurs et ne font donc pas l'objet de cette thèse :

— la récupération musculaire : c'est une interruption de travail assez longue pour permettre un relâchement des muscles impliqués dans l'activité de travail ;
— l'effort de préhension : cela représente l'effort à mettre en œuvre par l'agent de tri pour saisir les objets, en fonction de leur disposition sur la table, de leur taille, etc. ;
— le nombre d'objets triés : cela consiste à comptabiliser le nombre d'objets triés par l'opérateur ;
— quantité de matière triée : cela consiste à mesurer le poids total des objets triés par le valoriste.

Ces différents indicateurs et leur méthode de calcul sont donnés en Tableau 2.2.

2.2 Construction de notre système d'analyse automatique de la pénibilité

Les indicateurs ergonomiques étant maintenant établis, il reste à décrire le processus qui nous permet de les calculer à partir des images de profondeur. Ce calcul n'est pas direct, plusieurs étapes sont nécessaires avant d'être en mesure de calculer les indicateurs. Ces étapes sont présentées dans cette partie.

2.2.1 Calcul des indicateurs ergonomiques

On remarque rapidement que les indicateurs ergonomiques que nous souhaitons calculer sont des indicateurs qui peuvent directement être obtenus si l'on connaît la posture de l'opérateur de tri. En effet, en connaissant la position des articulations de l'opérateur de tri, on peut directement en calculer des angulations et donc établir la zone de risque dans laquelle évolue chaque articulation de la partie supérieure de l'opérateur.

Ainsi, l'obtention des indicateurs ergonomiques consiste finalement en un calcul géométrique direct effectué suite à une estimation de posture préalable.

2.2.2 Estimation de posture

Comme écrit précédemment, le calcul des indicateurs ergonomiques repose sur une estimation préalable de la posture 3D de l'opérateur. Une partie importante de cette thèse

Nom de l'indicateur	Quantité à mesurer	Mesure de la pénibilité
Évolution des membres supérieurs dans les zones à risque	Temps passé dans zones à risque	Faible < X% X% < Moyenne < Y% Y% < Élevée
Répétitivité des mobilisations musculaires	Occurrences dans zones à risque	Faible < X X < Moyenne < Y Y < Élevée
Maintien des postures à risques	Occurrences dans zones à risque d'une durée de D secondes ou plus	Faible < X X < Moyenne < Y Y < Élevée
Récupération musculaire	Temps passé dans une posture de repos	Faible > X% X% > Moyenne > Y% Y% > Élevée
Effort de préhension	Nombre de sollicitations utiles des poignets	Faible < X X < Moyenne < Y Y < Élevée
Mouvements par zone	Mouvements effectués dans chaque zone	Zone verte = X Zone jaune = Y Zone rouge = Z
Actions techniques	Nombre d'actions techniques	Faible < X X < Moyenne < Y Y < Élevée
Objets triés	Nombre d'objets triés	Faible > X X > Moyenne > Y Y > Élevée
Quantité triée	Poids de matière triée	Faible < X X < Moyenne < Y Y < Élevée

TABLE 2.2 – Les indicateurs ergonomiques finaux et leur méthode de calcul pour la mesure de la pénibilité de l'activité de l'agent de tri. Les valeurs D, X, Y et Z ne sont pas déterminées et sont propres à chaque indicateur.

consiste donc à établir un algorithme d'estimation de la posture du valoriste sur les images de profondeur.

Il est important de noter que nous ne pouvons pas utiliser les algorithmes d'estimation de posture déjà existants sur des images de profondeur, tels que celui proposé par la Kinect [Shotton et al., 2011], car ils ne donnent pas de résultats satisfaisants en contexte industriel pour les raisons suivantes :

— la qualité de l'estimation de posture dépend du point de vue de la caméra : à titre d'exemple, celle effectuée par la Kinect, qui est spécialement conçue pour une utilisation dans un cadre de divertissement, où le capteur est positionné face à l'utilisateur, ne donne pas de résultats satisfaisants lorsqu'on change sa position, en particulier à cause de sa méthode de calcul de la coordonnée de profondeur Z (dans les faits l'estimation de posture de la Kinect est de type 2D+Z) ;

— la présence de perturbations dues à l'environnement industriel, comme les bandes réfléchissantes sur les habits des opérateurs, détériore fortement la qualité de l'estimation.

— les occultations des membres des valoristes, très nombreuses lorsque le valoriste trie, que ce soit les auto-occultations ou bien celles causées par les objets à trier.

2.2.3 Segmentation des images de profondeur

Pour simplifier l'estimation de posture, un premier traitement à effectuer sur nos images de profondeur est d'en réaliser une segmentation. Cette segmentation a pour but d'isoler la silhouette de profondeur de l'opérateur de tri, afin que le reste de l'image de profondeur ne perturbe pas l'estimation. Des informations qui peuvent perturber l'estimation de la posture sont par exemple les objets tenus, la présence d'autres opérateurs autour de l'agent de tri, etc.

De plus, un autre argument en faveur de cette segmentation est que de nombreuses approches d'estimation de posture sur les images RVB se basent sur des imagettes où l'utilisateur dont on souhaite estimer la posture a déjà été détecté. De même, l'estimation de posture réalisée sur les images de profondeur par la Kinect [Shotton et al., 2011] est directement effectuée sur la silhouette de profondeur de l'utilisateur.

Il est donc primordial d'effectuer une segmentation de l'image de profondeur afin d'obtenir une silhouette de profondeur de l'agent de tri, afin de simplifier l'estimation de posture qui suit.

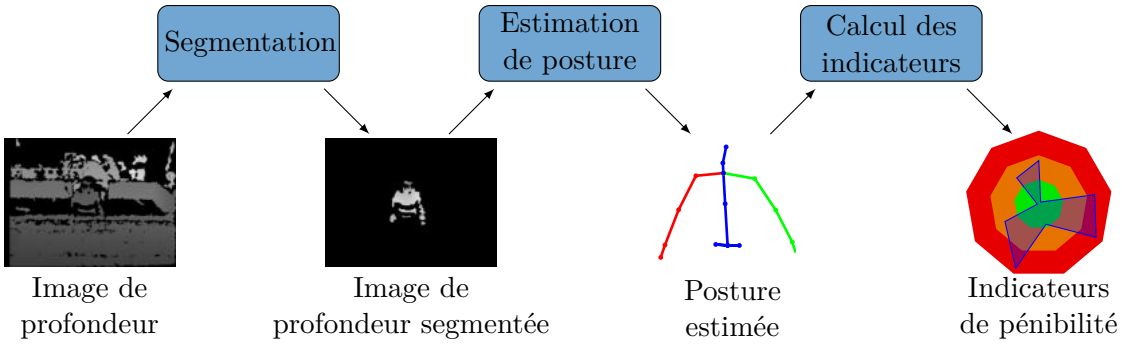


FIGURE 2.1 – Notre pipeline pour le calcul des indicateurs de pénibilité à partir des images de profondeur. La première étape consiste à extraire l'opérateur de l'image de profondeur grâce à une segmentation de celle-ci, puis la posture de l'opérateur est estimée et les indicateurs sont calculés à partir de la position des différentes articulations.

2.2.4 Pipeline final de notre processus de calcul des indicateurs de pénibilité

Ainsi, le calcul des indicateurs de pénibilité à partir des données de profondeur est divisé en trois étapes, comme le montre la Figure 2.1. Ces étapes sont les suivantes :

- une segmentation des images de profondeur afin d'isoler la silhouette de l'agent de tri ;
- une estimation de la posture de l'agent de tri effectuée sur l'image de profondeur segmentée ;
- une déduction par calcul géométrique des zones d'évolution des différentes articulations grâce à l'estimation de posture, puis un calcul des indicateurs.

En résumé

La norme *NF X35—702* ainsi qu'une étude menée en collaboration avec des ergonomes ont permis d'établir une liste d'indicateurs pertinents permettant d'estimer au mieux la pénibilité de l'activité de tri (Section 2.1). Ces indicateurs sont les zones d'évolution des membres supérieurs, les zones d'angulations pour les articulations des membres supérieurs, ainsi que des mesures temporelles de l'exposition à des zones à risque.

Connaissant les indicateurs à mesurer, nous avons pu construire un processus d'analyse automatique de la pénibilité (Section 2.2). Les indicateurs ergonomiques à mesurer étant des indicateurs pouvant directement être calculés à partir de la posture de l'opérateur, une première partie du travail sera de construire un algorithme capable de faire l'estimation de la posture, les performances des approches existantes étant limitées dans notre contexte industriel. Afin de simplifier cette tâche d'estimation de la posture, une segmentation de l'image de profondeur sera effectuée au préalable : elle permettra d'isoler la silhouette et les informations de profondeur qui concernent l'opérateur du reste de l'image de profondeur. Le pipeline final pour le calcul des indicateurs à partir des images de profondeur est donné en Figure 2.1.

Chapitre 3

État de l'art sur la segmentation d'image et l'estimation de posture

Sommaire

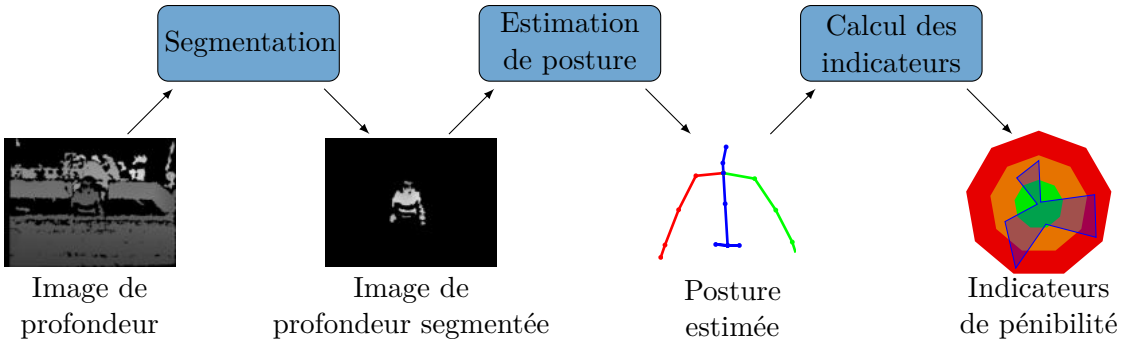


FIGURE 3.1 – Notre pipeline pour le calcul des indicateurs de pénibilité à partir des images de profondeur. La première étape consiste à extraire l'opérateur de l'image de profondeur grâce à une segmentation de celle-ci, puis la posture de l'opérateur est estimée et les indicateurs sont calculés à partir de la position des différentes articulations.

Dans ce chapitre, nous nous intéressons aux principales méthodes utilisées dans le cadre de la segmentation d'image et de l'estimation de posture humaine, en accord avec les différents modules à développer au cours de cette thèse (voir Figure 3.1). Nous commençons tout d'abord par brièvement introduire la problématique de la segmentation d'image, de l'estimation de posture et leurs applications. Comme de nombreuses approches ont recours à des algorithmes d'apprentissage automatique, en particulier pour l'estimation de posture humaine, nous proposons ensuite une description de l'apprentissage automatique, en nous concentrant sur l'apprentissage supervisé et l'apprentissage non supervisé, d'autres formes d'apprentissage étant abordées en Sections 4.2 et 6.2.1. Nous proposons dans une deuxième et troisième partie un état de l'art détaillé des principales approches traditionnelles pour la segmentation d'image et l'estimation de posture, les méthodes utilisant l'apprentissage profond étant décrites en Section 4.3 pour la segmentation et en Section 5.2 pour l'estimation de posture. Enfin, comme les travaux réalisés dans cette thèse s'appuient sur des modèles de réseaux de neurones, nous présentons le concept d'apprentissage profond ainsi que des mesures pour évaluer la qualité d'un apprentissage.

Segmentation d'image

La segmentation d'image est une tâche considérée comme très difficile en vision par ordinateur, en raison de la complexité que représente la classification de chaque pixel d'une image. L'objectif de la segmentation est de partitionner une image en régions significatives dans le but de simplifier la représentation d'une image et la rendre plus simple à analyser. Le résultat de la segmentation d'une image est un ensemble de régions disjointes qui recouvrent l'image ou bien un ensemble de contours qui séparent l'image en régions. Chacun des pixels de la même région ont des similarités par rapport à certaines caractéristiques (couleur, texture, intensité, localité, etc.). En revanche, les pixels qui n'appartiennent pas

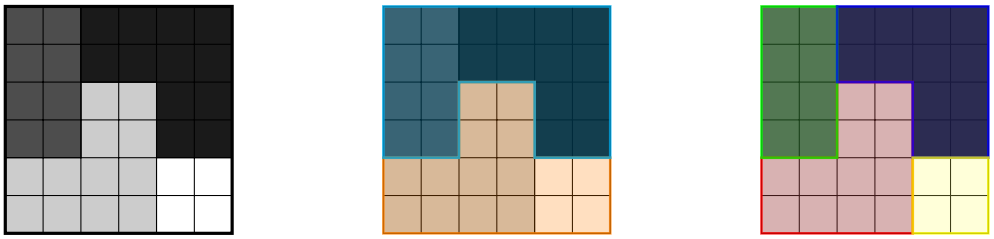


Image originale | Image segmentée en deux régions | Image segmentée en quatre régions

FIGURE 3.2 – Exemple de segmentation d'une image en régions. La segmentation peut ne pas être unique en fonction de ses paramètres, ici l'intensité de la nuance de gris.

à la même région ont des dissemblances par rapport à ces mêmes caractéristiques. Un exemple de segmentation est présenté en Figure 3.2.

Cette définition a été formalisée pour la première fois par Steven Horowitz et Theodosios Pavlidis [Horowitz, 1975, Horowitz and Pavlidis, 1976] :

Définition 3.0.1 (Segmentation) *Soit X le domaine d'une image. On définit un prédicat logique P sur les sous-ensembles S de X. La segmentation de X est une partition de X en sous-ensembles S_i, $i = 1, \ldots, m$, tel que :*

1. $X = \bigcup_{i=1}^{m} S_i$,
2. $S_i \bigcap S_j = \varnothing \ \forall i \neq j$,
3. $P(S_i)$ *est vrai* $\forall i$,
4. $P(S_i \bigcup S_j)$ *est faux* $\forall i \neq j$, *si* S_i *et* S_j *sont adjacents en* X.

Lorsqu'on analyse la Définition 3.0.1, on observe qu'une segmentation doit vérifier les critères suivants :

— 1. et 2. : la segmentation doit être une partition en régions disjointes de l'image originale : tout pixel appartient à une région et à une seule ;

— 3. : une région doit être homogène, c'est-à-dire que tous les pixels d'une même région doivent vérifier le prédicat ;

— 4. : une fusion de deux régions adjacentes ne doit pas être homogène, c'est-à-dire que chaque région de le segmentation doit être maximale.

Le prédicat peut porter sur différentes propriété de l'image. Par exemple, cela peut être une condition sur les écarts de couleur entre les pixels.

Les applications de la segmentation sont multiples. Parmi elles, on trouve par exemple :

— la recherche d'images basée sur le contenu [Carson et al., 2002, Lonarkar and Rao, 2017] ;

— l'imagerie médicale [Masulli and Schenone, 1999, Pham et al., 2000] pour l'aide au diagnostic [Fenster and Chiu, 2006, Alsmirat et al., 2017, Mejbri et al., 2019] ou à la chirurgie [Soler et al., 2001, Shvets et al., 2018] ;

— la détection d'objets [Leibe et al., 2008, Girshick et al., 2015], de piétons [Brazil et al., 2017, Ott and Everingham, 2009] et de visages [Albiol et al., 2001, Pujol et al., 2017] ;
— le contrôle qualité [Aydin et al., 2017] ;
— la vidéo-surveillance [Cheng and Butler, 2005].

Dans l'état de l'art qui suit, nous présentons les différentes approches traditionnelles pour la segmentation d'images et nous nous attardons sur le problème de la segmentation d'images de profondeur, problème qui sera ensuite traité par le Chapitre 4. Les approches utilisant l'apprentissage profond pour la segmentation d'images seront décrites en Section 4.3.

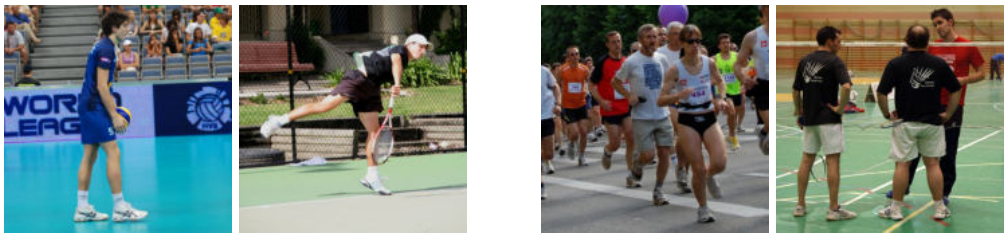
Estimation de posture

L'estimation de la posture humaine est la tâche consistant à localiser des points clés anatomiques sur des images, comme des parties du corps ou des articulations.

C'est un problème très difficile a résoudre, en raison de la complexité du corps humain :
— la variabilité d'apparence entre les personnes, qu'elle soit causée par des morphologies ou des tenues vestimentaires différentes, ainsi que les conditions d'acquisition, comme l'éclairage ou le point de vue, entraînent une grande variabilité d'observation pour la même posture ;
— à cause de ses nombreuses articulations, les modèles de corps humain ont de nombreux degrés de liberté, et la posture évolue dans un espace de grande dimension ;
— en fonction du mode d'observation, on peut avoir des ambiguïtés visuelles, avec en particulier une perte de l'information de profondeur de la scène lorsqu'on cherche à construire une estimation de posture en trois dimensions à partir d'une image (en deux dimensions) ;
— la structure complexe du corps humain et sa flexibilité provoquent souvent des auto-occultations, c'est-à-dire qu'une partie du corps en cache une autre. Les parties cachées sont difficiles à distinguer et donc leur position difficile à estimer (voir Figure 3.3, à gauche) ;
— lorsque l'on souhaite estimer la posture de plusieurs individus sur la même image, les parties du corps de certaines personnes peuvent occulter celles des autres (voir Figure 3.3, à droite).

L'estimation de posture a tout de même de nombreuses applications : elle permet par exemple le développement d'interface homme/machine sans marqueur [Stiefelhagen et al., 2004, Wachs et al., 2011], mais aussi un suivi des mouvements et une analyse bio-mécanique de l'utilisateur en temps réel [Corazza et al., 2006, Ray and Teizer, 2012]. Elle est souvent utilisée comme pré-traitement pour la reconnaissance d'action [Wang et al., 2013, Chéron et al., 2015].

La posture humaine peut être reconstruite avec précision à l'aide de marqueurs optiques



Auto-occultations Occultations inter-individus

FIGURE 3.3 – Exemples d'ambiguïtés visuelles. À gauche, on observe une auto-occultation, où des parties du corps des sportifs sont cachées par d'autres. À droite, des parties du corps de certains athlètes sont masquées par d'autres individus.

fixés sur les parties du corps de l'utilisateur. Cependant, ces systèmes ne sont pas adaptés à des applications non invasives et sont très coûteux, ce qui limite leur application à des expériences en laboratoire ou à des projets à très gros budget tels que l'animation de personnages de jeux vidéos ou la production de contenu cinématographique [Poppe, 2007b].

De nombreux travaux se sont donc focalisés sur l'estimation de posture sans marqueur, qui sont présentés dans cette partie. L'estimation de posture humaine étant un domaine de recherche très actif, de nombreux états de l'art couvrant le sujet ont ainsi été proposés [Perez-Sala et al., 2014, Liu et al., 2015b, Gong et al., 2016].

Le niveau de précision recherché dépend souvent de l'application et du système d'acquisition : pour certaines applications, telles que l'interaction homme machine, on ne cherche par exemple qu'à estimer la position de certains membres, comme la tête [Stiefelhagen et al., 2004] ou les mains [Nickel and Stiefelhagen, 2007], alors que des approches de suivi du mouvement demandent une estimation précise et complète de la posture [Andriluka et al., 2010]. En fonction du jeu de données et des contraintes, l'estimation peut ne concerner qu'une seule partie du corps [Lee and Cohen, 2004, Ferrari et al., 2008, Andriluka et al., 2009], souvent le haut du corps. Les travaux relatifs à l'estimation de posture présentent des méthodes d'estimation très variées, car ils dépendent de nombreux paramètres, comme par exemple la dimension de l'espace d'estimation (2D ou 3D), le nombre de postures à estimer (unique ou multiple), les données utilisées ou le nombre de capteurs utilisés.

Dans la suite de ce chapitre, nous présentons les différentes approches dites "classiques" de l'état de l'art pour l'estimation de posture sur des images. Nous nous concentrons en fin d'état de l'art sur l'estimation de posture à partir d'images de profondeur, problème qui sera ensuite traité en Chapitre 5. Les approches utilisant l'apprentissage profond pour l'estimation de posture seront décrites en Section 5.2.

La suite de ce chapitre est donc découpée en cinq parties. Tout d'abord, nous étudions les

concepts et principales méthodes d'apprentissage supervisé et non supervisé en Section 3.1. Ensuite, en accord avec le pipeline illustré en Figure 3.1, nous proposons un état de l'art des méthodes traditionnelles de segmentation d'images en Section 3.2, puis d'estimation de posture en Section 3.3. Enfin, comme les travaux développés dans les chapitres suivants s'appuient sur des modèles de réseaux de neurones, nous présentons l'apprentissage profond en Section 3.4 et des mesures pour évaluer la qualité d'un apprentissage en Section 3.5.

3.1 Apprentissage automatique

L'apprentissage automatique (en anglais *machine learning*) est un domaine d'étude de l'intelligence artificielle fondé sur les algorithmes et des modèles statistiques pour donner la capacité à des ordinateur d'effectuer une tâche spécifique sans utiliser d'instruction explicite, en les faisant "apprendre" à réaliser ces tâches à partir de données appelées "données d'entraînement" [Hastie et al., 2009]. Les algorithmes d'apprentissage automatique construisent donc un modèle mathématique à partir de ces données, afin de faire des prédictions ou prendre des décisions.

L'apprentissage automatique se découpe communément en deux phases :

— une première phase d'entraînement, qui consiste à construire de manière automatique un modèle à partir d'observations ;
— une seconde phase de production, où le modèle entraîné pendant la phase précédente est utilisé pour résoudre la tâche pour laquelle il a été élaboré. Il est possible de continuer à entraîner le modèle pendant la phase de production.

Selon les informations utilisées pendant l'étape d'entraînement, le type d'apprentissage est différent. On peut distinguer trois principaux types d'apprentissage :

— si on possède des données annotées (ou étiquetées), c'est-à-dire que pour chaque donnée on connaît le résultat attendu, on parle d'apprentissage supervisé ;
— si l'on ne possède que des données, mais qu'elles ne sont pas annotées, on parle d'apprentissage non supervisé : l'algorithme doit alors trouver par lui-même la structure sous-jacente des données pour résoudre le problème qui lui est posé ;
— parfois, le modèle à entraîner peut représenter un agent décisionnel qui doit apprendre à effectuer des actions dans un environnement afin d'optimiser les récompenses qu'il reçoit dans cet environnement. On parle alors d'apprentissage par renforcement. Ce type d'apprentissage n'est pas présenté ici, car il n'est généralement pas utilisé pour résoudre les problèmes abordés au cours de cette thèse.

D'autres types d'apprentissage peuvent être dérivés de ces trois principales catégories : par exemple, si l'on possède à la fois des données annotées et des données non étiquetées, on peut combiner un apprentissage supervisé sur les données étiquetées et un apprentissage non supervisé sur les données non annotées. On dit alors que l'apprentissage est semi-supervisé. Ce type d'apprentissage est détaillé dans le Chapitre 6.

Dans cette partie, nous présentons les deux principaux types d'apprentissage : l'apprentissage supervisé (Section 3.1.1) et l'apprentissage non supervisé (Section 3.1.2). Nous proposons une description des approches classiques qui ont été utilisées dans l'état de l'art de la segmentation d'images ou l'estimation de posture, ou encore dans nos travaux, et notre objectif n'est pas d'être exhaustif.

3.1.1 Apprentissage supervisé

L'apprentissage supervisé (en anglais *supervised learning*) est un domaine de l'apprentissage automatique qui consiste à construire un modèle à partir d'exemples annotés, c'est-à-dire que pour chaque donnée, on peut associer une réponse attendue. Ces exemples annotés forment un ensemble d'apprentissage composé de paires constituées d'une donnée d'entrée et d'une valeur de sortie désirée [Russell and Norvig, 2002]. On suppose que cette base de données est représentative d'un ensemble d'échantillons plus large, dans le but que le modèle soit capable d'être général, c'est-à-dire capable de donner des résultats corrects sur des données qui ne font pas partie de l'ensemble d'apprentissage.

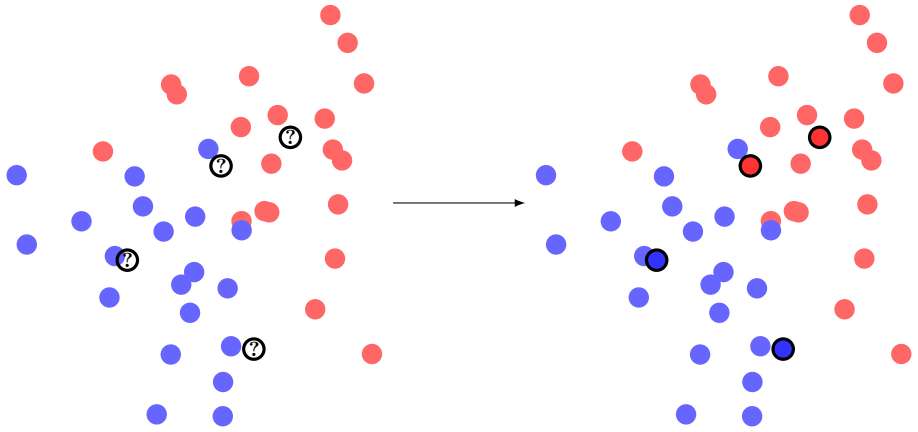
L'apprentissage supervisé peut être utilisé pour résoudre des problèmes de classification, si les étiquettes sont discrètes (par exemple dire si une image contient un chat ou un chien [Golle, 2008]), ou de régression, si les étiquettes sont continues (comme estimer la position d'un objet [Covell et al., 2006]).

À partir de ces données, on peut entraîner différents types d'algorithmes, dont certains sont représentés par la Figure 3.4. Ces méthodes d'apprentissage supervisé sont basées sur :

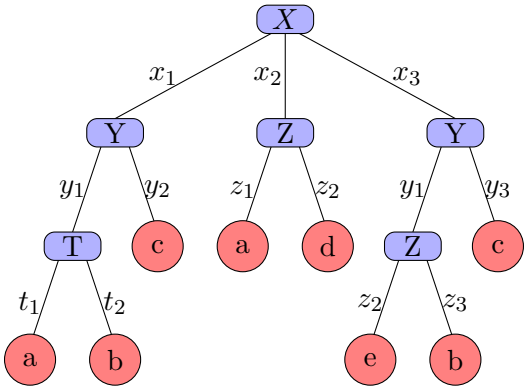
— la **distance entre les données** dans l'espace euclidien comme la méthode des k plus proches voisins (abrégée k-NN de l'anglais *k-nearest neighbors*) qui se sert de la proximité entre les données d'entrée pour prédire la sortie d'entrées inconnues ;

— la **densité de probabilité** comme la classification bayésienne reposant sur une forte indépendance des caractéristiques et permettant de définir un classifieur bayésien grâce au calcul du maximum de vraisemblance.

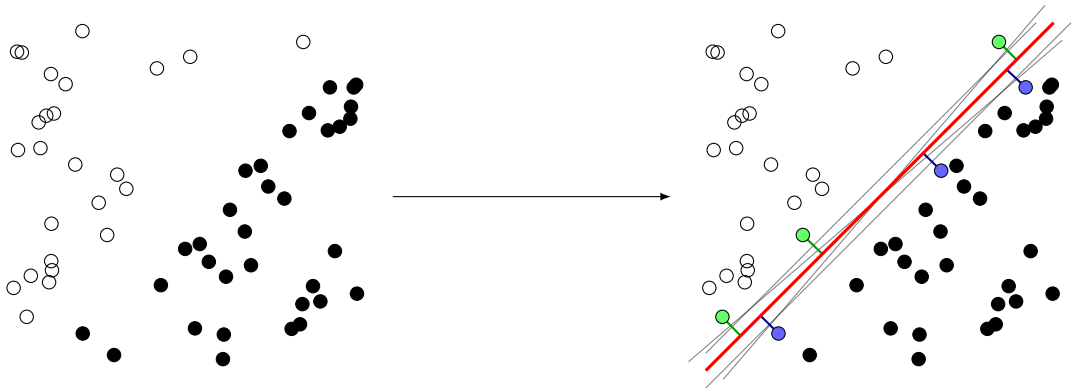
— la **représentation par arbre des données** avec les arbres décisionnels construits en séparant récursivement les données en sous-ensembles en fonction de la valeur d'une caractéristique. Les forêts d'arbres décisionnels (en anglais *Random Decision Forests*), version généralisée des arbres décisionnels, combine la sélection aléatoire de caractéristiques [Amit and Geman, 1997] et le *bagging* [Breiman, 1996]. Ce dernier permet de générer plusieurs versions d'un même prédicteur et de les utiliser ensemble pour obtenir un prédicteur agrégé. L'agrégation consiste à faire la moyenne des résultats des différents prédicteurs (ici des arbres de décision) lorsqu'on souhaite un résultat numérique, et à un vote lors de la prédiction d'une classe. Cette méthode est conçue pour augmenter la robustesse par rapport au bruit et éviter un



(a) Exemple d'application de la méthode des k plus proches voisins pour la classification avec $k = 3$. La classe associée aux nouveaux points est la classe majoritaire parmi les trois plus proches voisins de ces points.



(b) Exemple d'arbre de décision pour la classification. Le test de certaines caractéristiques (en bleu) d'un échantillon, permet de descendre dans l'arbre en suivant la valeur de la caractéristique testée jusqu'à arriver à une feuille qui donne la classe de l'échantillon (en rouge). De mêmes caractéristiques peuvent être testées sur des branches différentes et différentes feuilles peuvent représenter la même classe.



(c) Exemple illustratif de machine à vecteurs de support. Les deux classes différentes sont représentées par les points blancs et les points noirs. La séparatrice optimale est donnée en rouge et ses vecteurs de support sont affichés en bleu et vert. Des exemples de séparatrices non-optimales sont affichés en gris.

FIGURE 3.4 – Exemples de méthodes d'apprentissage supervisé. On y retrouve les k plus proches voisins (a), l'arbre de décision (b) et la machine à vecteurs de support (c).

surapprentissage.

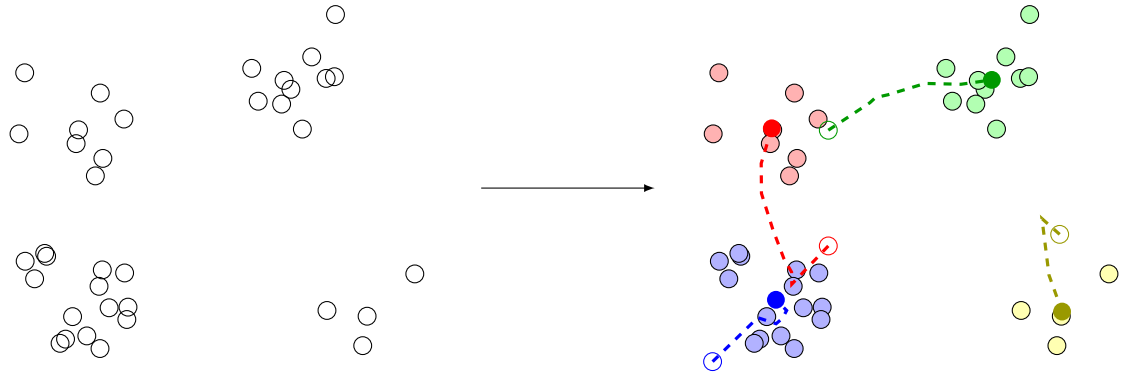
— la **prise en compte des dépendances entre voisins** avec les champs aléatoires de Markov (en anglais *Markov random fields*, abrégé MRF), qui est un ensemble de variables aléatoires vérifiant une propriété de Markov par rapport à un graphe non orienté. On retrouve aussi les champs aléatoires conditionnels (en anglais *conditional random fields*, abrégé CRFs) [Lafferty et al., 2001], qui sont une catégorie d'approches de modélisation statistique utilisée en particulier pour des données séquencées comme le texte. Alors qu'un classifieur prédit généralement une classe pour un échantillon sans prendre en compte les échantillons voisins dans la séquence, un champ aléatoire conditionnel cherche à tenir compte du contexte. Ainsi, la prédiction est modélisée sous la forme d'un graphe, qui représente les dépendances entre les prédictions d'échantillons voisins. L'apprentissage des paramètres est généralement réalisé par optimisation du maximum de vraisemblance.

— la **définition d'un séparateur** comme les machines à vecteurs de support (abrégé SVM, de l'anglais *support vector machines*) [Vapnik, 1995, Cortes and Vapnik, 1995]. Ces séparateurs à vaste marge cherchent à maximiser la marge entre les frontières qui séparent les différentes classes et les échantillons les plus proches. Ces échantillons sont appelés vecteurs supports. Ce sont des classifieurs linéaires mais le recours à l'"astuce du noyau" (en anglais *kernel trick*) permet de généraliser le type de séparation en ayant recours à un espace de redescription de dimension supérieure via une fonction noyau (de type polynomiale, exponentielle ou gaussienne). Dans cet espace, les données peuvent être linéairement séparables. Il existe aussi une extension des machines à vecteurs de support, les *Relevance Vector Machines* [Tipping, 2001], qui utilisent l'inférence bayésienne pour la classification et la régression.

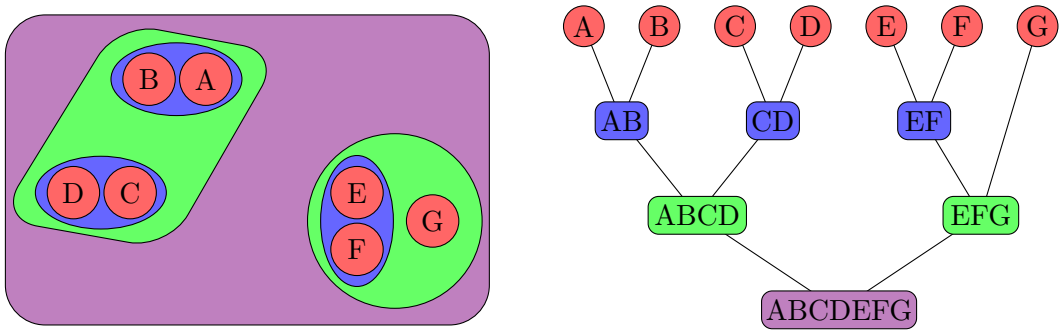
3.1.2 Apprentissage non supervisé

À la différence de l'apprentissage supervisé, l'apprentissage non supervisé (en anglais *unsupervised learning*) construit un modèle à partir de données non annotées, et avec un minimum de supervision humaine. L'apprentissage non supervisé permet donc de modéliser des densités de probabilité sur les entrées [Hinton et al., 1999]. Il cherche en particulier à repérer des structures naturellement présentes dans les données. Ainsi, plutôt que de chercher à prédire une information une information à partir des données, l'apprentissage non supervisé analyse les données dans leur globalité pour en extraire une information générale, telle que leur distribution.

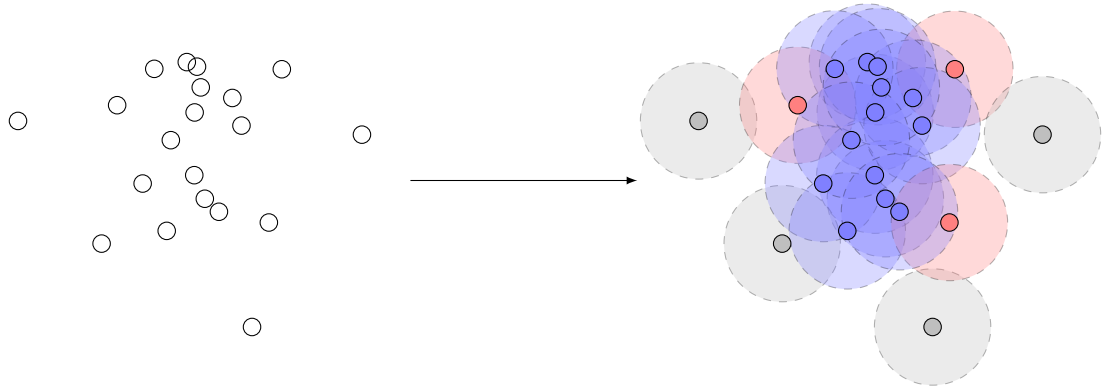
Les principales méthodes utilisées pour l'apprentissage supervisé, avant la popularisation des réseaux de neurones, sont les algorithmes de partitionnement de données. Dans cette partie, nous présentons certaines approches utilisées dans le cadre de la segmentation d'image et/ou de l'estimation de posture.



(a) Exemple de partitionnement d'un ensemble de points en quatre classes par l'algorithme des k-moyennes. Les centroïdes ont été initialisés aléatoirement et leur trajectoire au cours de l'algorithme est indiquée en pointillés.



(b) Exemple de regroupement hiérarchique, où l'on utilise la distance euclidienne comme mesure de dissimilarité. Pour un regroupement hiérarchique ascendant, on part de groupes individuels (en rouge), puis on construit la hiérarchie en fusionnant les groupes (en bleu, puis en vert et enfin en violet). Dans le cadre d'un partitionnement hiérarchique descendant, on part d'un groupe unique (en violet), qu'on divise récursivement (groupes verts, puis bleus et enfin rouges).



(c) Exemple d'application de l'algorithme DBSCAN. La distance maximale est représentée par les aires colorées, et le nombre minimal de voisins est fixé à trois. On obtient un unique groupe, dont les points centraux sont affichés en bleu et les points frontières en rouge. Les point gris sont des points n'appartenant à aucun groupe, c'est-à-dire des points considérés aberrants.

FIGURE 3.5 – Exemples de méthodes d'apprentissage non supervisé par partitionnement. On retrouve le partitionnement par k-moyennes (a), le partitionnement hiérarchique (b) et l'algorithme DBSCAN (c).

Ces méthodes, dont certaines sont illustrées en Figure 3.5, sont basées sur :

— les **distances géométriques** entre les données dans l'espace euclidien comme le partitionnement par k-moyennes (en anglais *k-means*), illustré en Figure 3.5a, proposé par [Steinhaus, 1956], dont le but est de diviser un ensemble de points en k groupes, de manière à minimiser une fonction prédéfinie (en général la variance des données autour de leur centroïde assigné) ou bien comme le partitionnement hiérarchique [Ward Jr, 1963], présenté en Figure 3.5b, qui cherche à établir une hiérarchie entre les groupes par une mesure de dissimilarité ;

— la **modélisation des données par des distributions de probabilité** comme DBSCAN (pour *density-based spatial clustering of applications with noise*) [Ester et al., 1996], illustré en Figure 3.5c, reliant les points qui satisfont un critère de densité, défini par un nombre de points minimums inclus dans un rayon. Dans le même esprit en utilisant des fonctions noyaux, la méthode *Mean-shift* [Cheng, 1995, Comaniciu and Meer, 2002] cherche à localiser les maxima, appelés "modes", de la fonction de densité qui représente les échantillons discrets ;

— la **modélisation des données par un graphe non orienté** en considérant les données comme les sommets et le poids des arêtes comme la distance entre les données. Le partitionnement est réalisé en coupant le graphe via des mesures de coupes (*Graph cut* [Boykov and Jolly, 2001, Boykov and Funka-Lea, 2006], dont une application pour la segmentation est présentée en Figure 3.11, ou *Normalized Cut* [Shi and Malik, 2000]) basées notamment sur le poids des arêtes intra et inter classes et le cardinal des classes.

3.2 Segmentation d'image

Avant que les réseaux de neurones ne deviennent incontournables, la segmentation d'image pouvait être réalisée à l'aide d'un grand nombre d'approches. Nous présentons ici les principales méthodes de la littérature.

3.2.1 Segmentation basée sur le seuillage

Les méthodes les plus simples de segmentation sont basées sur le seuillage : on choisit une ou plusieurs valeurs seuils qui permettent de partitionner l'image en fonction de la valeur de ses pixels. Elles sont décrites ici.

Seuillage sur histogramme

Dans la méthode proposée par Otsu [Otsu, 1979], utilisée pour binariser des images, on utilise l'histogramme pour trouver la valeur seuil qui minimise la variance intra-classe, car minimiser la variance intra-classe revient à maximiser la variance inter-classe.

Méthodes de regroupement

Une technique semblable à l'idée du seuillage est de faire appel à des algorithmes de regroupement tels que les k-moyennes [Ray and Turi, 1999], de *mean shift* [Comaniciu and Meer, 2002, Zheng et al., 2009], Quick Shift [Vedaldi and Soatto, 2008], ou encore SLIC (pour *simple linear iterative clustering*) appliqués directement sur les pixels de l'image ou sur leur histogramme, qui permettent de construire des superpixels, dont on peut, pour certains algorithmes, contrôler le nombre.

3.2.2 Segmentation basée sur les régions

Les méthodes de segmentation basées sur les régions manipulent directement des régions. L'objectif est d'identifier des zones dont les pixels possèdent des propriétés communes. Il existe plusieurs types d'approches qui sont décrites ici.

La segmentation par découpe de régions (en anglais *split*)

Les segmentations qui appliquent la division de régions partent de l'échelle la plus grossière pour arriver vers les détails les plus fins. L'initialisation peut être faite en utilisant toute l'image ou seulement une région prédéfinie que l'on souhaite raffiner. Le procédé de ces méthodes est itératif : on divise les régions en sous-régions qui sont elles-même subdivisées, jusqu'à atteindre des régions dont les pixels sont homogènes. [Ohlander et al., 1978] utilise par exemple les histogrammes des niveaux de gris des régions pour réaliser la division. Les résultats sont donnés sous forme d'arbres (exemple en Figure 3.6), souvent quaternaires (en anglais *quadtree*), c'est-à-dire des arbres pour lesquels chaque nœud a quatre fils. Ce type de méthodes est très minoritaire et elles ne sont souvent que la première étape d'une approche combinant une découpe puis une fusion des régions (ces approches sont présentées plus loin).

La segmentation par fusion de régions (en anglais *merging*)

Ces segmentations utilisent la fusion de sous-régions à partir d'une pré-segmentation (en général une sur-segmentation) [Peng et al., 2011]. Elles font parfois appel à un utilisateur pour déterminer le nombre de classes final [Ning et al., 2010]. Ces sous-régions sont itérativement regroupées jusqu'à obtenir des classes homogènes en optimisant un certain critère, combinant souvent similarité colorimétrique et proximité spatiale, qui vérifient la Définition 3.0.1 d'une segmentation. À l'inverse de la segmentation par découpe de région, un problème de cette approche est le risque de sous-segmentation.

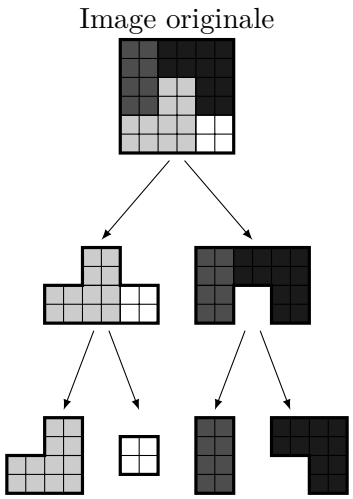


FIGURE 3.6 – Exemple d'arbre représentant les segmentations obtenues en Figure 3.2. Cette représentation expose une structure hiérarchique entre les régions. On remarque qu'il est possible de construire plusieurs segmentations, en fonction de la profondeur parcourue sur chaque branche.

Image segmentée en quatre régions

Graphe d'adjacence

FIGURE 3.7 – Exemple de graphe d'adjacence pour la segmentation obtenue. Les sommets du graphe représentent les différentes régions obtenues. Les arêtes connectent les régions adjacentes. Ce graphe est particulièrement utile pour effectuer l'étape de fusion dans des approches découpe/fusion.

La segmentation par découpe et fusion (en anglais *split and merge*)

Cette méthode a été introduite par la première fois en 1974 par Horowitz [Horowitz, 1975], dans l'optique de corriger la sur-segmentation issue de la découpe de régions. L'idée est de combiner les segmentations par découpe et par fusion de régions présentées précédemment : on effectue tout d'abord une découpe en régions de l'image, puis on fusionne certaines régions obtenues, jusqu'à obtenir une segmentation qui vérifie la Définition 3.0.1. Lors de la fusion, on se sert de la représentation hiérarchique en structure d'arbre obtenue lors de la découpe (voir Figure 3.6) et d'un graphe d'adjacence (voir Figure 3.7) qui expose les relations de proximité entre les différentes régions.

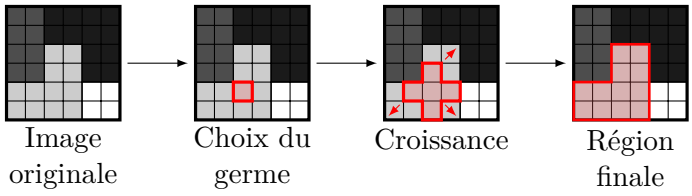


FIGURE 3.8 – Exemple de segmentation d'une image par croissance de région.

La segmentation par croissance de région (en anglais *region growing*)

Un type de segmentation populaire est la segmentation par croissance de région [Adams and Bischof, 1994, Pohle and Toennies, 2001]. Celle-ci utilise des pixels de départ appelés "germes" (en anglais *seed*) et les étend en ajoutant les pixels ou les régions du voisinage qui satisfont le critères de similarité. Sinon, une nouvelle région est créée. On peut choisir les germes manuellement ou par une méthode automatique [Espindola et al., 2006]. Le programme de segmentation interactive GrowCut [Vezhnevets and Konouchine, 2005] fait appel à un algorithme de croissance de région, les utilisateurs donnant les germes et pouvant influer sur la segmentation tout au long du processus pour guider l'algorithme. Il est important de noter que le choix des critères de similarité a une influence sur la segmentation finale.

La segmentation par ligne de partage des eaux (en anglais *watershed*)

Ce concept a été introduit en 1979 par Serge Beucher et Christian Lantuéjoul [Beucher and Lantuéjoul, 1979], puis appliqué à la segmentation d'images par Serge Beucher en 1992 [Beucher, 1992]. Cette approche voit une image comme une carte topographique, avec un paramètre définissant la hauteur de chaque pixel (par exemple la norme du gradient). Il s'agit alors de calculer la ligne de partage des eaux de ce relief. Les bassins versants correspondent ainsi aux régions de la segmentation. Ils existe trois types d'algorithmes pour la construction de la ligne de partage des eaux : les algorithmes par inondation (voir Figure 3.9), qui simulent une montée progressive du niveau de l'eau à partir des minima de la carte topographique, les algorithmes de ruissellement, où l'eau part d'un pixel et suit la ligne de gradient maximal (c'est-à-dire de plus grande pente) jusqu'à atteindre un minimum et les algorithmes proposant une déformation progressive de la topologie jusqu'à le réduire en une structure correspondant aux lignes de partage des eaux. Un défaut de cette approche est cependant l'obtention de sur-segmentations : on peut avoir autant de régions que de minima locaux.

3.2.3 Segmentation basée sur les contours

Ces approches exploitent le fait qu'il existe une transition détectable entre deux régions connexes. L'objectif de ces méthodes est de segmenter l'image en détectant les contours et

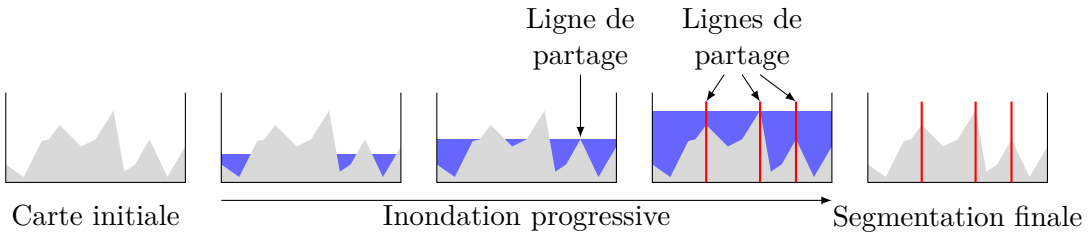


FIGURE 3.9 – Exemple de segmentation par ligne de partage des eaux suivant une approche d'inondation progressive de la carte topologique d'un signal en une dimension. L'eau monte progressivement et lorsque deux bassins se rejoignent, une ligne de partage des eaux est marquée. Les régions séparées par ces lignes de partage constituent la segmentation finale.

en les reliant pour former des courbes fermées délimitant des régions, ou bien en raffinant des contours annotés de manière grossière par un utilisateur.

La segmentation par détection des contours

Ces méthodes sont divisées en deux étapes et se basent sur la détection de contour pour construire des régions fermées.

Tout d'abord, on applique des algorithmes de détection des contours [Dhankhar and Sahu, 2013] : le filtre de Canny [Canny, 1986], de Sobel [Sobel and Feldman, 1968] ou de Prewitt [Prewitt, 1970], qui donnent une estimation de la dérivée directionnelle de l'image par rapport à un axe. Un contour peut aussi être vu comme un maximum local du gradient dans l'image avec l'utilisation du laplacien par la détection directe de ses passages par zéro. Ces filtres sont cependant sensibles au bruit et il peut être nécessaire de réaliser un prétraitement : au lieu d'utiliser directement le laplacien, on peut utiliser le filtre LoG (*Laplacian of Gaussian*) [Qian and Huang, 1996], où le noyau gaussien a pour but de lisser l'image.

La seconde étape consiste à affiner les contours : tous les contours ne servent pas à délimiter des régions. Pour ne conserver que des contours significatifs, on effectue généralement un seuillage à hystérésis [Medina-Carnicer et al., 2011]. De plus, afin de connecter les contours qui ne sont parfois ni fermés ni continus, il est nécessaire d'utiliser une méthode de suivi [Sumengen and Manjunath, 2005] ou de fermeture des contours.

La segmentation par contours déformables

La segmentation par contours déformables, aussi appelés contours actifs ou *snake*, a été introduite par Kass et Witkin en 1987 [Kass et al., 1987]. Cette approche utilise un contour initial qui est progressivement modifié de manière à épouser le contour recherché. Pour effectuer la transformation de ce contour, on applique principalement deux types de forces : les forces internes, qui contrôlent la régularité de la courbe de contour, et les forces

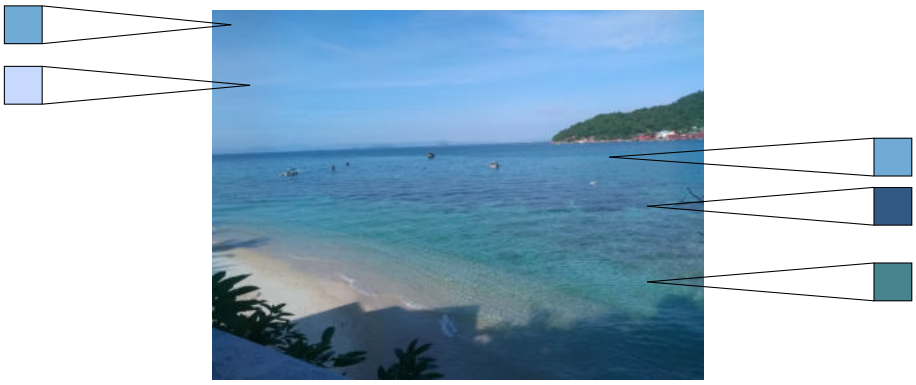


FIGURE 3.10 – Exemple d'image où l'information globale est importante pour construire une segmentation sémantique. En effet, certains pixels du ciel et de la mer ont la même couleur, alors que d'autres, appartenant pourtant à la même classe, ont des couleurs très différentes. Pour les distinguer, il faut donc ajouter à cette information locale une information plus globale.

externes, qui sont liées aux propriétés de l'image. Ces méthodes ont l'avantage d'être peu sensibles au bruit et sont souvent employées dans le contexte de l'imagerie médicale [Yezzi et al., 1997, Delingette and Montagnat, 2001, Zimmer et al., 2002].

3.2.4 Segmentation par apprentissage supervisé

Avant que les réseaux de neurones n'inondent de résultats le domaine de la segmentation d'images, quelques approches faisant appel à de l'apprentissage supervisé ont été proposées.

La segmentation d'images par apprentissage supervisé est souvent une segmentation sémantique, c'est-à-dire qu'en plus de partitionner l'image en régions, l'algorithme associe une classe à chacune d'entre elles. La raison pour laquelle la grande majorité des approches de segmentation par réseaux de neurones préfère la segmentation sémantique à la segmentation "traditionnelle" vient du fait que les apprentissages supervisés sont construits pour des problèmes de classification.

La segmentation sémantique d'image est un problème complexe dans le sens où il nécessite la prise en compte de la localité de chaque pixel (un petit champ de vue), pour obtenir une segmentation précise, combinée à une compréhension globale de l'image (un large champ de vue). Aussi, un des enjeux majeurs des apprentissages supervisés en segmentation d'images est d'être capable de fusionner l'information locale et le contexte global dans une même architecture (voir Figure 3.10).

Les algorithmes d'apprentissage supervisé pour la segmentation d'images utilisent en général les caractéristiques suivantes :

— la **couleur des pixels** : elle peut être traitées dans différents espaces comme les niveaux de gris, l'espace RVB, l'espace TSV (pour Teinte Saturation Valeur, en

anglais *HSV*) ou l'espace TSI (pour Teinte, Saturation, Intensité, en anglais *HSI*) ;

— les **descripteurs SIFT** (pour *Scale-Invariant Feature Transform*) [Lowe, 1999], traduisible par "transformation de caractéristiques invariante à l'échelle", construits de manière à effectuer une analyse locale d'une image de la manière la plus indépendante possible de l'éclairage, de l'angle de vue, du cadrage et de la résolution. Cette analyse permet la détection de points d'intérêt à plusieurs échelles, à l'aide d'une pyramide de l'image à différentes résolutions, où les extrema d'une différence de gaussiennes permettent d'obtenir les points d'intérêt. ;

— l'**histogramme de gradient orienté** (en anglais *histogram of oriented gradient* ou HOG) [Dalal and Triggs, 2005] décrit l'orientation locale et l'intensité du gradient et donc des contours. Il est calculé en divisant l'image en cellules de petite taille, sur lesquelles on calcule un histogramme des directions du gradient. En combinant ces histogrammes locaux, on obtient le descripteur HOG (voir Figure 3.15) ;

— les **sac de mots visuels** (en anglais *Bag-of-visual-words* ou BOV) [Csurka et al., 2004] : les sacs de mots visuels sont des descripteurs locaux de l'image. L'image est représentée par un histogramme des fréquences d'apparition de ces descripteurs ;

— des **caractéristiques obtenues par réduction de la dimension**, comme par exemple après une analyse en composantes principales (ACP) [Pearson, 1901] ;

— les **textons** [Julesz, 1981] : il n'y a pas de définition formelle, mais un texton est la composante de base minimale de la vision. Ce sont des éléments de base pour la perception des textures.

Les forêts de décision aléatoire (en anglais *Random Decision Forests*) [Ho, 1995] ont été utilisées pour la segmentation d'images par [Shotton et al., 2008]. Les auteurs utilisent des arbres de décision qui travaillent directement sur les pixels de l'image. Ces arbres fournissent un regroupement hiérarchique implicite en textons sémantiques et une estimation explicite de la classification globale. Les textons sémantiques sont combinés sous la forme de sac de textons sémantiques, c'est-à-dire des histogrammes de textons calculés sur l'ensemble de l'image. Ces caractéristiques permettent de construire une segmentation sémantique de l'image.

Des algorithmes basés sur les machines à vecteur de support ont été proposés : [Felzenszwalb et al., 2009] définit un système pour la détection d'objets utilisant une machine à vecteur de support. Cette machine est utilisée sur des caractéristiques dérivées des histogrammes de gradient orientés, calculées à différentes résolutions sous la forme d'une pyramide. La machine à vecteur de support est entraînée à l'aide d'une descente de gradient stochastique. Les résultats des segmentations sont donnés sous la forme de boîtes englobantes, consruites d'après les sorties de la segmentation réalisée par la machine à vecteur de support.

Les champs aléatoires de Markov ont aussi servi pour la segmentation d'images. [Zhang et al., 2001] utilise un modèle de champ aléatoire de Markov caché entraîné à l'aide d'un

algorithme d'espérance-maximisation (en anglais *expectation maximisation*) [Dempster et al., 1977]. Ils montrent que cette approche est robuste vis-à-vis du bruit.

Enfin, certaines méthodes font appel aux champs aléatoires conditionnels (en anglais *conditional random fields* ou CRFs) [Lafferty et al., 2001]. [He et al., 2004] propose une approche pour inclure des caractéristiques contextuelles pendant le calcul de la segmentation. Le modèle utilisé combine différents composants qui s'intéressent à différentes informations : un classifieur est dédié aux statistiques locales de l'image, un autre aux motifs locaux, un se concentre sur les annotations locales et un autre sur des annotations globales grossières. Ces classifieurs sont intégrés dans un champ aléatoire conditionnel. Ils sont entraînés grâce au critère de probabilité maximale conditionnelle. Une approche basée sur les textons est à nouveau proposée par [Shotton et al., 2006]. Cette fois-ci, le modèle combine une information sur l'apparence, la forme et le contexte. Le modèle exploite des caractéristiques construites sur les textons, qui modélisent à la fois la forme et la texture. La segmentation de l'image est obtenue en incorporant ces classifieurs à un champ aléatoire conditionnel.

Ces caractéristiques sont intégrées dans un cadre probabiliste représenté par des champs aléatoires conditionnels multi-résolution.

3.2.5 Autres méthodes de segmentation

Enfin, quelques approches graphiques ou basées sur les méthodes variationnelles ont été proposées. Elle sont présentées ici.

Segmentation basée sur l'utilisation de graphes

Le problème de la segmentation d'image a aussi été posé sous la forme d'un problème d'optimisation combinatoire. En effet, une image peut être vue comme un graphe non-orienté dont les sommets sont les pixels et les arêtes représentent les relations de voisinage entre les pixels [Boykov and Jolly, 2001]. On peut dont segmenter l'image en rompant les liens entre les sommets au niveau des arrêtes. Une segmentation optimale est trouvée pour une coupe de coût minimale par rapport aux propriétés intégrées dans le poids des arêtes. Deux techniques populaires sont Graph Cut [Boykov and Jolly, 2001, Boykov and Funka-Lea, 2006] (voir Figure 3.11), dont une version interactive a été proposée par [Heimann et al., 2004], ou Normalized Cut [Shi and Malik, 2000]. Il est important de noter que ces méthodes sont généralement interactives afin d'établir les poids des arêtes pour réaliser la meilleure segmentation. Cette approche est souvent utilisée en segmentation d'objet, pour séparer une forme de l'arrière-plan.

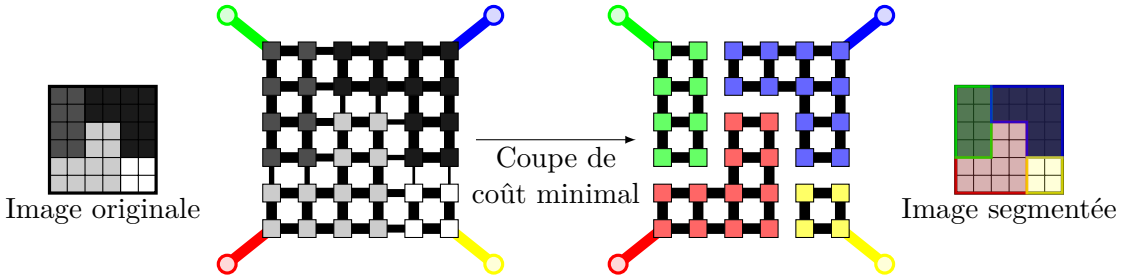


FIGURE 3.11 – Exemple d'application de coupe de graphe pour la segmentation. Le poids des arêtes est fonction de la similarité de couleur et est indiquée par l'épaisseur du trait.

La segmentation par approches variationnelles

La segmentation peut être vue comme un problème de minimisation d'énergie : l'objectif est d'effectuer une minimisation globale sous contraintes entre l'image originale et l'image segmentée. Les méthodes variationnelles peuvent donc être utilisées dans le but de minimiser une fonctionnelle qui mesure l'écart entre l'image segmentée et l'image originale [Tsai et al., 2001]. Les approches les plus courantes cherchent à minimiser la fonctionnelle de Mumford-Shah [Mumford and Shah, 1989, Pock et al., 2009] : l'objectif est de trouver une image résultat qui soit proche de l'image originale mais constante par morceaux, les parties constantes étant les régions composant la segmentation de l'image. Ces contraintes se formalisent de la manière suivante [Strekalovskiy and Cremers, 2014] :

Définition 3.2.1 *Soit l'image* $i_0 : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$. *La segmentation* $s : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ *de* i_0 *est la fonction qui vérifie :*

$$\min_{s:\Omega\subset\mathbb{R}^2\to\mathbb{R}} \iint_{(x,y)\in\Omega} |s(x,y) - i_0(x,y)|^2 + \lambda\delta(||\nabla s(x,y)||)dxdy \tag{3.1}$$

où :

- $|s(x,y) - i_0(x,y)|^2$ *mesure la différence entre* i_0 *et* s *(attache aux données) ;*
- $\lambda\delta(||\nabla s(x,y)||)$ *mesure la constance par morceaux (régularisation).*

Ce problème peut-être résolu par des approches basées sur la descente de gradient [Tsai et al., 2001] ou des approximations [Ambrosio and Tortorelli, 1990].

3.2.6 Segmentation d'images de profondeur

Il existe peu de travaux traitant de la segmentation d'images de profondeur, car le capteur le plus couramment utilisé, c'est-à-dire la Kinect de Microsoft, fournit déjà une segmentation appropriée de l'utilisateur dans les cas d'utilisation courants (c'est-à-dire lorsque le capteur se trouve face à l'utilisateur).

FIGURE 3.12 – Exemple de segmentation de l'utilisateur sur des images de profondeur. Images extraites du dataset NTU RGB+D pour la reconnaissance d'actions [Shahroudy et al., 2016]. Lorsque l'utilisateur tient un objet ou est proche d'un meuble, celui-ci peut être considéré comme faisant partie de la silhouette de l'utilisateur et provoquer des erreurs de segmentation.

De plus, dans le cas général, un seuillage ou un regroupement effectué grâce à l'information de profondeur est suffisant [Bryant and Pettigrew, 2015] : les objets ou les utilisateurs sont souvent situés à des distances différentes du capteur de profondeur, et ne sont pas non plus agglutinés les uns contre les autres. Ils peuvent donc facilement être distingués les uns des autres.

Des difficultés peuvent malgré tout survenir lorsque les utilisateurs sont en contact entre eux ou avec un objet : les utilisateurs sont difficilement séparés et les objets sont identifiés comme faisant partie intégrante de la silhouette de l'utilisateur. Les objets étant généralement de petite taille, cela n'impacte souvent pas significativement les performances des post-traitements. Un cas parfois difficile à gérer survient lorsque l'utilisateur est assis ou allongé : en fonction de la position, le mobilier peut ne pas être séparé de la forme de l'utilisateur (voir Figure 3.12).

La segmentation d'images de profondeur peut néanmoins aussi être vue comme un problème de segmentation de nuage de points. En effet, certains capteurs de profondeur fournissent l'information de profondeur comme un nuage de point, et pour les autres, il est possible de transformer l'image de profondeur en un nuage de points, si l'on connaît les paramètres intrinsèques du capteur. Ce nuage de points peut ensuite être segmenté. Dans cette thèse, nous ne traitons les données de profondeur que comme des images et nous ne nous sommes pas intéressés aux travaux touchant au traitement de nuages de points. Cependant, une revue récente de l'état de l'art pour la segmentation de nuages de points a été proposée par [Xie et al., 2020]. On y retrouve des approches similaires à celles proposées pour la segmentation d'images, adaptées pour être appliquées aux nuages de points, comme par exemple PointNet [Qi et al., 2017b] ou PointNet++ [Qi et al., 2017a].

3.3 Estimation de posture humaine

Les travaux portant sur l'estimation de la posture humaine font suite à ceux portant sur la détection de piétons, tels que les travaux précurseurs de Hogg [Hogg, 1983] ou la célèbre approche proposée par Viola et Jones [Viola et al., 2005].

On distingue deux types de méthodes pour l'estimation de posture à partir d'images :

— les approches basées sur des **modèles** : les travaux proposés s'appuient sur un modèle défini a priori du corps humain, dont on souhaite retrouver la configuration ;

— les techniques fondées sur des **exemples** : ici, l'estimation de la posture est réalisée directement, mais l'algorithme se sert d'un ensemble de données contenant des couples d'images et de la posture associée ;

Dans cette partie, nous présentons tout d'abord les différentes caractéristiques observées sur les images pour réaliser l'estimation de la posture, puis les différents types d'approches existantes.

3.3.1 Caractéristiques utilisées

Les approches d'estimation de la posture, peu importe la technique utilisée, doivent être capables d'analyser l'image afin d'en extraire des informations à différents niveaux et échelles qui permettent de déduire une estimation haut niveau de la posture humaine présentée sur l'image. Elles s'appuient donc sur différentes caractéristiques extraites de l'image.

Ces caractéristiques doivent être rapides et simples à retrouver, et présenter une certaine robustesse vis à vis des conditions dans lesquelles les images ont été acquises, tout en décrivant assez d'information pour analyser l'image de manière efficace [Gong et al., 2016].

Les caractéristiques utilisées pour l'estimation de posture sont présentées dans cette partie.

Caractéristiques de bas niveau

Pour capturer correctement l'apparence, la geométrie et des informations sur la forme globale de l'humain ou des parties de son corps, certaines caractéristiques de bas niveau peuvent être extraites des images (illustrées en Figure 3.13) :

— la **silhouette** : la silhouette peut être calculée de manière robuste et précise si la caméra est statique et que le fond de la scène varie peu. Elle a l'avantage d'être invariante aux textures et aux conditions d'éclairage [Agarwal and Triggs, 2005]. Une suppression de l'arrière plan peut être suffisante pour calculer une silhouette de l'individu présent au premier plan. L'extraction de silhouette construit un masque binaire qui distingue la personne du reste de l'image [Rosales and Sclaroff, 2000]. C'est un descripteur global de la posture, qui contient une grande quantité d'information. Mais la silhouette rend invisible certains degrés de liberté de la posture, et

Image originale | Silhouette | Contours

FIGURE 3.13 – Exemples de caractéristiques de bas niveau. À gauche, on a l'image originale. Au milieu, on a extrait la silhouette de l'individu, et à droite, on a extrait les contours de l'image.

des postures différentes peuvent avoir la même silhouette [Agarwal and Triggs, 2005], en partie à cause d'une ambiguité de symétrie. La présence de bruit et d'artefacts peut poser problème pour l'estimation de la posture, les silhouettes extraites peuvent donc être post-traitées pour s'en débarasser [Sminchisescu and Telea, 2002] ;

— les **contours** : les contours permettent d'extraire un aperçu des parties du corps en découpant l'image en régions. Ils peuvent être calculés à partir de filtres de convolution [Dimitrijevic et al., 2006], tels que les caractéristiques pseudo-Haar [Viola and Jones, 2001], ou bien résulter d'une segmentation de l'image [Sapp et al., 2010b]. Pour filtrer ces contours qui ne sont parfois pas utiles à l'estimation de posture, certaines approches ne considèrent que les contours situés à l'intérieur de la silhouette [Shakhnarovich et al., 2003].

— la **couleur** et les **textures** : on les utilise en faisant l'hypothèse que la couleur des parties du corps et la texture des vêtements portés par l'individu ne changent pas, dans ce cas, les caractéristiques sont donc plutôt utilisées lorsqu'on traite des vidéos [Pfister et al., 2015], ou lorsqu'on dispose de plusieurs vues [Hofmann and Gavrila, 2012]. Certaines approches utilisent la texture pour séparer l'arrière plan du premier plan, où se situe la personne [Mori et al., 2004].

Caractéristiques de haut niveau

Les caractéristiques de bas niveau sont généralement transformées en caractéristiques de plus haut niveau, telles que :

— le **descripteur *shape context*** [Belongie et al., 2001] : ce descripteur est construit sur des points du contour d'une forme, ici l'individu, échantillonnés de manière assez uniforme [Belongie et al., 2002]. Pour chaque point de l'échantillon, on construit un histogramme en deux dimensions de la position en coordonnées polaires logarithmiques des autres points du contour par rapport à celui-ci (voir Figure 3.14). Pour le problème de l'estimation de posture, on utilise le contour d'une forme associée à une posture, que l'on déforme progressivement jusqu'à ce que les descripteurs *shape*

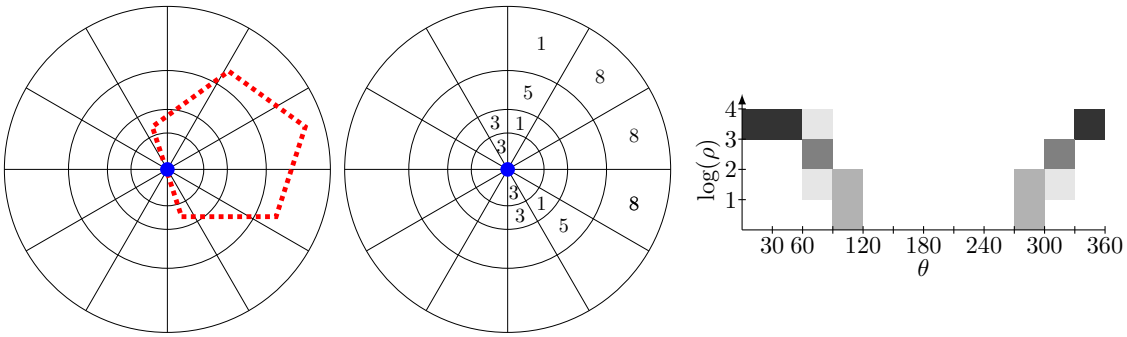


FIGURE 3.14 – Exemple de construction d'histogramme pour le calcul du descripteur *shape context*. Dans le repère polaire logarithmique ayant pour origine le point dont on calcule l'histogramme (à gauche), on compte le nombre de points du contour dans chaque zone (au milieu) et on construit l'histogramme associé (à droite).

Image originale Descripteur HOG

FIGURE 3.15 – Exemple de descripteur HOG obtenu en appliquant l'histogramme de gradient orienté à une image. Le zoom sur certaines parties du corps permet d'observer la structure du descripteur, où l'on retrouve les variations de forme, de couleur et de texture de l'image.

context des deux formes soient assez proches [Mori et al., 2004]. La posture du contour déformé est donc la posture estimée. Le descripteur *shape context* a été étendu en *generalized shape context*, qui prend en compte l'orientation des contours [Mori et al., 2005] ;

— l'**histogramme de gradient orienté** [Dalal and Triggs, 2005] (abrégé HOG de l'anglais *histogram of oriented gradients*) a été utilisé par [Yang and Ramanan, 2011] pour calculer la configuration locale des parties ;

— les **descripteurs SIFT** [Lowe, 1999, Lowe, 2004], dont l'algorithme a été protégé par un brevet juqu'au 9 mars 2020. Certaines approches pour l'estimation de postures antérieures utilisent des descripteurs "proches" des descripteurs SIFT, c'est-à-dire calculés de la même manière [Agarwal and Triggs, 2006], ou une extension en trois dimensions (deux spatiales et une temporelle)[Scovanner et al., 2007] pour la reconnaissance d'actions ;

— les ***poselets*** [Bourdev and Malik, 2009] : ce sont des éléments qui permettent cha-

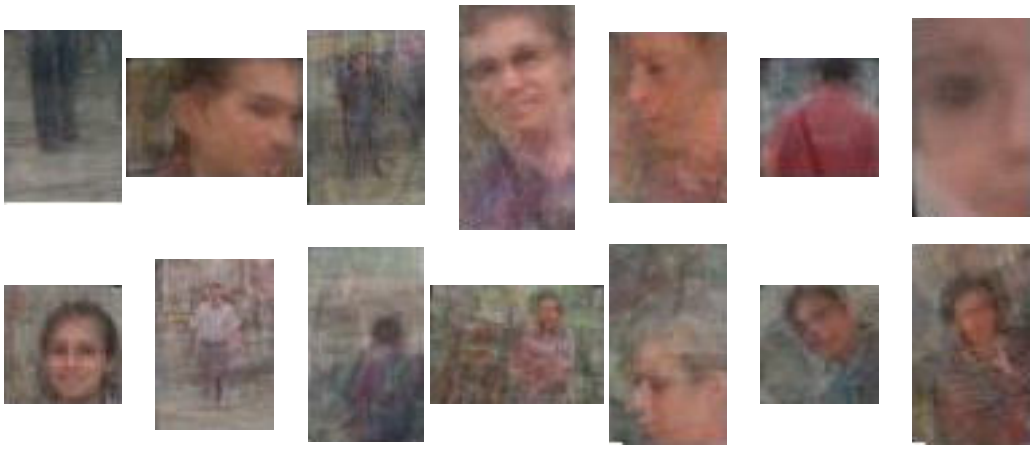


FIGURE 3.16 – Exemples de *poselet*. Les *poselets* incarnent différentes parties du corps dans des configurations variables. Certains *poselets* ont une granularité très fine, alors que d'autres représentent une grande partie du corps.Images issues des travaux de [Bourdev and Malik, 2009].

cun de décrire une partie précise de la posture sous un point de vue donné. Ce sont des patchs représentant une partie du corps, allant d'un membre jusqu'au corps complet, dans une certaine position (voir Figure 3.16). Un poselet peut être détecté dans une image en recherchant son histogramme de gradient orienté dans celui de l'image [Wang et al., 2011]. Ce sont des caractéristiques qui capturent la configuration des différentes parties du corps ;

— d'autres **descripteurs hiérarchiques**, tels que HMAX, les *hyperfeatures*, la pyramide spatiale, les arbres de vocabulaire et les *multilevel spatial blocks* ont été comparés pour l'estimation de posture semi-supervisée [Kanaujia et al., 2007]. HMAX [Serre et al., 2005] est un modèle multicouche inspiré de l'anatomie du cortex visuel qui permet de reconstruire des représentations de plus en plus invariantes à la résolution et à la translation quand on parcourt les niveaux, grâce à des opérations de *max-pooling*. Les *hyperfeatures* [Agarwal and Triggs, 2006] sont une version plus homogène de HMAX, qui utilise la moyenne plutôt que le maximum. La pyramide spatiale [Lazebnik et al., 2006] construit des histogrammes sur des régions de l'image de plus en plus grandes. L'arbre de vocabulaire [Nister and Stewenius, 2006] est un modèle hiérarchique basé sur un partitionnement des données à l'aide de l'algorithme des k-moyennes. Les groupes des différents niveaux sont construits de manière récursive depuis les niveaux plus grossiers vers les niveaux les plus fins. Les *multilevel spatial blocks* [Kanaujia et al., 2007] sont des niveaux consistant en une grille régulière de descripteurs SIFT de patchs d'images se chevauchant avec une taille de patch croissante ;

— les **caractéristiques temporelles** : lorsque l'on possède des données sous forme

de vidéo ou lorsque l'on souhaite effectuer une estimation de posture à l'aide d'un suivi, on peut utiliser des caractéristiques temporelles comme le flux optique [Horn and Schunck, 1981]. Le flux optique permet de modéliser le mouvement des objets à l'image au cours du temps, et donc de suivre l'évolution de la posture au cours du temps [Ju et al., 1996]. Des modèles permettent de mesurer la relation entre le mouvement mesuré dans l'image et le mouvement de la posture [Bregler and Malik, 1998]. Certaines approches utilisent des descripteurs plus robustes, tels que les frontières de mouvement et l'énergie du contour, qui mesurent la modification de la position des contours amenée par les mouvements [Sminchisescu and Triggs, 2001].

3.3.2 Méthodes basées sur des modèles

Pour représenter l'individu analysé, certaines approches s'appuient sur l'utilisation d'un modèle du corps humain, défini explicitement et a priori. Le traitement de l'image permet de calculer la configuration du modèle. Les modèles utilisés sont variés (voir Figure 3.17) et les approches utilisent une fonction de vraisemblance, parfois appelée fonction de compatibilité [Yang and Ramanan, 2011] qui est optimisée pour maximiser la cohérence entre l'image observée et la posture estimée. Cette fonction est généralement coûteuse à optimiser : elle est évaluée sur les poses prédites jusqu'à trouver la meilleure configuration du modèle, en tenant généralement compte de l'apparence et de la structure spatiale du modèle. On peut par exemple chercher à ajuster les contours du modèle à ceux de l'image [Delamarre and Faugeras, 2001] ou utiliser la distance de chanfrein (*Chamfer distance*) [Gavrila and Davis, 1996]. Ces méthodes fonctionnent généralement à l'aide d'une boucle d'"analyse-synthèse" [Ning et al., 2008], c'est-à-dire qu'une première configuration du modèle est prédite, puis elle est itérativement mise à jour à l'aide des informations contenues dans l'image.

Dans cette partie, nous présentons différents modèles qui permettent de représenter le corps humain, puis des approches d'estimation de la posture basées sur un modèle.

Les différents types de modèles utilisés

Un des paramètres les plus importants dans les travaux d'estimation de posture à l'aide d'un modèle est la définition de ce modèle. En effet, ce modèle doit être construit de manière à contenir des informations sur la structure cinématique du corps humain, et éventuellement sur sa forme et son apparence. Comme la modélisation est explicite, on peut introduire des contraintes sur modèles comme par exemple une proportionnalité entre la taille des différentes parties du corps [Wang et al., 2014] ou des angulations maximales pour les articulations [Akhter and Black, 2015].

En général, plus un modèle est complexe, plus le résultat peut être précis, mais plus le

Modèle cinématique | Modèle volumétrique | Modèle planaire

FIGURE 3.17 – Les modèles pouvant être utilisés pour l'estimation de posture. À gauche, un exemple de modèle cinématique d'humain composé de vingt-et-une articulations, au milieu, un maillage 3D représentant un individu, et, à droite, un modèle planaire, où les différentes parties du corps sont représentées par des morceaux de plan.

modèle est complexe, plus l'estimation est coûteuse. La complexité d'un modèle se mesure au nombre de paramètres à calculer, qui représentent les degrés de liberté du modèle.

On peut distinguer trois types de modèles :

— les modèles basés sur des **ensembles de parties du corps** (un exemple est donné en Figure 3.17, à droite) : ce sont des modèles utilisés pour une estimation de posture par la détection, appropriés à l'estimation de la posture en 2D. Les approches qui les utilisent cherchent en effet à détecter la position de différentes parties du corps tout en conservant une configuration globale cohérente et plausible. Dans une approche présentées par [Ju et al., 1996], le modèle est appelé "*card-board person model*" et est constitué d'un ensembe de patchs planaires connectés représentant les membres du corps. Les structures picturales (en anglais *pictorial structures*) [Fischler and Elschlager, 1973], qui sont des modèles assez similaires, ont été très utilisées [Andriluka et al., 2010, Sapp et al., 2010a, Yang and Ramanan, 2011]. Elles consistent en un ensemble de parties liées dont la structure est représentée a priori par un graphe ou plusieurs graphes, souvent sous la forme d'arbres [Wang and Mori, 2008, Andriluka et al., 2009], et les contraintes entre les parties sont modélisées par des distributions gaussiennes appliquées dans un espace où chaque partie du corps est représentée par sa position, son orientation et son échelle [Felzenszwalb and Huttenlocher, 2005]. L'estimation de posture par structures picturales généralisées en 3D est proposée par [Sigal et al., 2012]. Des modèles utilisant des grammaires [Girshick et al., 2011b] ont aussi été utilisés. Des règles de compositions permettent de représenter le corps humain comme une combinaison de ses parties (le tronc, les bras, les jambes et le visage), pouvant elles-mêmes être construites comme un assemblage d'éléments (le visage est par exemple composé des yeux, du nez et de la bouche) [Perez-Sala

et al., 2014]. On trouve par ailleurs dans cette catégorie de modèle, des approches qui recourent aux *poselets* [Wang et al., 2011], où le corps est représenté comme une combinaison hiérarchique de ses parties. Un modèle d'estimation de posture en 3D pour le suivi est proposé par [Sigal et al., 2004] : c'est un graphe ou chaque nœud correspond à une partie du corps, définie par sa position et son orientation en trois dimensions, alors que les arêtes modélisent les relations de position et d'angle entre les parties du corps adjacentes dans l'espace, mais aussi dans le temps.

— les modèles fondés sur la **structure du corps humain** (un exemple est donné en Figure 3.17, à gauche) : la structure du squelette humain étant très similaire à celle d'un arbre, la plupart des modèles basés sur la structure du corps humain sont représentés sous la forme de chaînes cinématiques organisées sous la forme d'un arbre. Par opposition aux modèles présentés précédemment, les nœuds de ces arbres représentent des points importants de la structure humaine, en général des articulations (épaule, coude, etc.). Ces modèles sont particulièrement adaptés à l'estimation de posture en trois dimensions [Perez-Sala et al., 2014]. Les contraintes cinématiques représentées par ce modèle permettent en effet de réduire l'ambiguité inhérente à l'estimation d'une posture en trois dimensions à partir d'une image. Le nombre de degrés de liberté de ces modèles peut varier considérablement en fonction du nombre de nœuds de la structure cinématique, et certains modèles ont plus de cinquante degrés de liberté [Barrón and Kakadiaris, 2001].

— les modèles représentant le **volume du corps humain** (voir Figure 3.17, au milieu) : ils permettent de modéliser l'apparence du corps humain en trois dimensions [Sidenbladh et al., 2000]. Certaines approches ajoutent aux squelettes cinématiques une représentation volumétrique des membres par des morceaux de cônes [Deutscher et al., 2000] ou des ellipsoïdes [Sminchisescu and Triggs, 2003]. Certaines méthodes recourent à des modèles de maillages 3D [De Aguiar et al., 2007], comme le modèle SCAPE (pour *Shape Completion and Animation of PEople*) [Anguelov et al., 2005], utilisé par [Balan et al., 2007] et [Guan et al., 2009]. Ce modèle a été amélioré par le modèle de la *stitched puppet* (que l'on peut traduire par "marionnette cousue") [Zuffi and Black, 2015], qui introduit un "coût de couture" pour l'élongation des parties du corps. Ce type de modèle a, par ailleurs, été utilisé pour l'estimation de posture en deux dimensions, par l'utilisation de modèles basés sur le contour [Freifeld et al., 2010].

Estimation de la posture

Résoudre le problème de l'estimation de posture avec une méthode basée sur un modèle consiste à maximiser une fonction mesurant la qualité de la configuration choisie : cette optimisation permet d'obtenir la configuration du modèle qui correspond le mieux à l'image analysée. On parle alors d'approche générative. Pour maximiser cette fonction, on utilise

des outils habituels d'optimisation, tels que la descente de gradient [Ju et al., 1996] ou un algorithme d'espérance-maximisation [Ménier et al., 2006]. L'espace des postures étant un espace de très grande dimension, l'optimisation est généralement difficile, car elle présente souvent un grand nombre d'optima locaux [Sminchisescu and Telea, 2002]. Ainsi, certaines approches basées sur le suivi utilisent l'estimation réalisée sur l'image précédente pour estimer la posture sur l'image courante [Bregler and Malik, 1998], ou utilisent plusieurs vues consécutives [Gavrila and Davis, 1996].

Quelques approches ont cherché à modéliser de manière plus fidèle les relations entre les différentes parties du corps, en plus de leurs liens structuraux de connectivité. Par exemple, l'approche proposée par [Lan and Huttenlocher, 2005] étend l'arbre des relations entre les parties du corps en ajoutant des liaisons pour tenir compte de la coordination des membres lors de la pratique d'activités telles que la marche ou la danse. Cette amélioration permet de résoudre de nombreuses erreurs engendrées par des cas d'auto-occultations, récurrents lorsque l'on analyse ce type d'activité. La méthode introduite dans [Pishchulin et al., 2013] ajoute elle aussi des contraintes à son modèle, basée sur des *poselets* pour prendre en compte la coordination entre les membres. Une méthode présentée par [Wang and Mori, 2008] utilise plusieurs modèles représentant chacun un type de relation différent entre les parties, pour réaliser une unique prédiction.

D'autres approches ont cherché à étendre la structure picturale en :

— proposant de partitionner l'espace des postures et de construire une structure picturale par partition [Johnson and Everingham, 2010]. Les différentes expositions d'une partie du corps sont ainsi distinguées et des machines à vecteurs support sont ensuite entraînées pour construire un détecteur de fenêtre glissante. Cette approche est améliorée par [Johnson and Everingham, 2011], à l'aide de données annotées par production participative, donc peu précises et contenant des erreurs qui sont par la suite classées par qualité et par type ou supprimées ;

— présentant une structure picturale multi-résolution [Sapp et al., 2010b] en augmentant progressivement la résolution de l'image. Le niveau grossier permet de filtrer l'espace des postures pour accélérer le calcul des structures picturales suivantes. [Sapp and Taskar, 2013] améliore cette méthode avec un modèle décomposable de posture humaine et d'un partitionnement de l'espace des postures ;

— proposant un modèle décomposable de posture humaine et d'un partitionnement de l'espace des postures [Yang and Ramanan, 2013]. Ce modèle mélange les relations entre les positions des parties et les relations de co-occurrence entre les parties en apprenant conjointement tous les paramètres. Les résultats sont meilleurs que ceux dont la position de chaque partie du corps est estimée séparément.

En général, les performances des méthodes d'estimation de posture basées sur des modèles reposent sur la capacité de celui-ci de capturer l'apparence du corps humain [Perez-Sala et al., 2014] pour la détection des parties du corps. Cependant, de nombreux faux

positifs peuvent être trouvés, et l'utilisation de contraintes spatiales modélisant la structure du corps humain permet de corriger ces erreurs. Ainsi, les meilleures approches d'estimation de la posture sont celles dont le modèle représente les différentes parties du corps associées à leur structure.

3.3.3 Méthodes basées sur des exemples

Par rapport aux approches basées sur des modèles, les approches qui s'appuient sur des exemples fournissent directement une estimation de la posture 3D à partir de l'image, sans chercher à optimiser la configuration d'un modèle explicitement défini. Elles se servent d'une base de données constituée de couples image/posture associée. Les données peuvent être obtenues par annotation manuelle [Johnson and Everingham, 2010, Andriluka et al., 2014], l'utilisation d'un dispositif de capture de mouvement [Sigal et al., 2010], ou bien à partir de logiciels d'animation et de rendu 3D, comme Blender [1] [Blender Online Community, 2020]. Il n'y a pas de modélisation explicite du corps humain, c'est-à-dire qu'on ne décrit pas l'apparence du corps humain ni ses contraintes cinématiques, mais ces informations sont implicitement fournies dans les postures de la base de données.

Un avantage important de ces méthodes est qu'elles n'ont pas besoin d'effectuer le rendu d'un modèle du corps humain, et qu'elles évitent de passer par l'optimisation d'une fonction de vraisemblance coûteuse à effectuer. Elles réalisent en général la prédiction d'une posture à partir d'une seule image, sans avoir besoin d'initialisation. Cependant, leur capacité de généralisation étant souvent assez limitée [Gong et al., 2016], il est nécessaire que l'ensemble de données utilisé présente une variabilité suffisante, afin de contenir des postures similaires à celles que l'on souhaite prédire.

On distingue ici deux types d'approches :

— les approches basées sur la comparaison aux exemples, où la base de données sert de référence pour estimer la posture sur de nouvelles données, et est stockée en mémoire pour servir de base de comparaison ;

— les approches fondées sur un apprentissage, où un entraînement est effectué afin d'optimiser les paramètres d'une fonction d'estimation de posture. Les approches d'estimation de la posture par apprentissage profond font partie de ces approches (voir Section 5.2).

Ces deux types de méthodes pour l'estimation de posture basée sur les exemples sont présentés dans cette partie.

Approches fondées sur la comparaison aux exemples

Dans ces méthodes, l'espace des posture est représenté par un ensemble fini de postures, qui représentent l'ensemble des postures possibles. Ainsi, chaque posture est associée au

1. https://www.blender.org

descripteur de son image (des exemples de descripteurs sont présentés en Section 3.3.1). Pour estimer la posture d'une nouvelle image, on compare son descripteur à ceux des images de l'ensemble de données, et on récupère la posture dont le descripteur est le plus proche. Certaines approches proposent d'interpoler la posture à partir des n descripteurs les plus proches : une posture est interpolée à partir des vingt-cinq postures candidates les plus proches par [Poppe and Poel, 2006], où les auteurs comparent différents descripteurs, puis par [Poppe, 2007a] en utilisant l'histogramme de gradient orienté calculé dans la boite englobante contenant la silhouette de l'individu.

Les auteurs de [Mori and Malik, 2006] utilisent les *shape contexts* appliqués aux contours de la silhouette de la personne. L'exemple le plus proche dans la base de données est ensuite déformé pour représenter plus fidèlement la silhouette recherchée et faire correspondre sa posture à la posture présentée sur l'image.

Dans une méthode proposée par [Chen et al., 2011], les descripteurs des images dont on souhaite estimer la posture sont exprimés comme des combinaisons linéaires creuses des descripteurs des images de la base de données. On récupère ainsi un groupe de postures candidates pour chaque image, puis on applique un algorithme de programmation dynamique dans le but de reconstruire une séquence continue.

Cependant, la recherche des plus proches voisins a deux inconvénients majeurs : la quantité d'espace mémoire nécessaire pour stocker la base de données d'exemples et le temps de calcul pour la comparaison entre les descripteurs lors de la recherche des exemples les plus proches. [Shakhnarovich et al., 2003] introduit le *Parameter-Sensitive Hashing*, qui permet d'apprendre des fonctions de hachages qui permettent une indexation efficace en fonction de la proximité entre les descripteurs à l'aide de classificateurs binaires. Les plus proches voisins servent ensuite à réaliser une régression locale pondérée pour l'estimation de la posture finale. Ceci permet à l'approche proposée d'utiliser un ensemble de données composé de 150 000 images et posture construit à l'aide du logiciel POSER [2].

[Jiang, 2010] utilise une base de données de plusieurs millions d'exemples pour reconstruire une posture 3D à partir de la posture 2D fournie associée à une image, en séparant la posture du haut du corps de celle du bas du corps. Pour circuler dans la base de données, les auteurs utilisent un arbre k-d (en anglais *k-d tree* pour *k-dimensional tree*) et atteignent une performance temps réel.

Méthodes basées sur l'apprentissage

D'autres méthodes utilisent les données comme un ensemble d'apprentissage pour entraîner un modèle paramétrique capable d'associer une posture à une image. Les approches d'estimation de posture par apprentissage profond (voir 5.2) font généralement partie de ce type de méthodes. Plusieurs manières de construire le modèle d'estimation existent.

Une première façon consiste à construire une fonction qui relie l'image ou un de ses

2. https://www.posersoftware.com/

descripteurs à la posture qui lui correspond : une régression permet d'estimer une posture directement à partir de l'image. Les différentes approches se différencient par le choix du descripteur utilisé. Une approche utilisant les silhouettes réalise, à l'aide d'une *relevance vector machine*, qui combine un modèle appris à des descripteurs de forme extraits de la silhouette, l'estimation de posture 3D [Agarwal and Triggs, 2004a], puis le suivi 3D d'un humain [Agarwal and Triggs, 2004b]. Une autre méthode proposée par les même auteurs [Agarwal and Triggs, 2005] utilise le descripteur *shape context* pour estimer la posture 3D. Les auteurs comparent plusieurs méthodes de régression comme les machines à vecteurs de support ou les *relevance vector machines*. L'approche de [Okada and Soatto, 2008] découpe l'espace des postures en différentes classes, puis se sert de l'histogramme de gradient orienté pour effectuer une première classification de l'image par machines à vecteur de support, puis une régression linéaire par morceaux pour estimer la posture 3D finale en fonction des postures appartenant à la même classe. L'histogramme de gradient orienté est aussi utilisé par [Agarwal and Triggs, 2006] pour apprendre des caractéristiques locales du corps humain pour l'estimation de posture sur des images encombrées, dans des cas où la silhouette ne peut pas être extraite de l'image, tels que les contours des épaules. [Sminchisescu et al., 2005] modélisent la distribution conditionnelle de la posture 3D par une distribution de probabilité multi-modale à l'aide d'un mélange bayésien de fonctions d'approximation appelé *bayesian mixture of experts model* [Jordan and Jacobs, 1994], apprise sur le descripteur *shape context* des silhouettes d'un ensemble de données de synthèse générées à l'aide du logiciel Maya[3]. Une approche proposée par [Sigal and Black, 2006] utilise également un mélange d'experts pour inférer la posture 3D à partir de la posture 2D. Les forêts aléatoires (voir Section 3.1.1) ont aussi été utilisées pour l'estimation de posture. Elles sont utilisées comme régresseur de points d'intérêts de la posture 2D par [Dantone et al., 2013]. Les forêts opèrent ici sur deux couches : la première couche sert à classer les parties du corps, alors que la seconde estime la position des points en tenant compte des prédictions de la première couche, pour que l'ensemble des parties du corps ait une influence sur l'estimation de la position de chaque point. Une approche proposée par [Yasin et al., 2016] utilise les forêts aléatoires pour estimer la posture 2D, puis retrouve la posture 3D à l'aide d'un arbre k-d de paires de postures 2D/3D. La posture 3D est ensuite raffinée pour corriger les erreurs d'estimation de la posture 2D.

Une seconde approche consiste à modéliser l'espace des postures : cet espace est représenté par la variété qui contient l'ensemble des configurations possibles du corps humain. Ce type de méthode a été introduit par [Brand, 1999], où la variété est modélisée par une chaine de Markov cachée et apprise par minimisation d'entropie. Pour une séquence donnée, l'algorithme calcule la trajectoire dans l'espace des postures 3D qui correspond le mieux à la dynamique apprise pour donner une estimation de la posture 3D à chaque image. [Elgammal and Lee, 2004] propose une approche qui utilise une variété par acti-

3. https://www.autodesk.fr/products/maya/overview

vité ou point de vue. Ces variétés sont apprises à partir des données d'entraînement sous forme de silhouettes, puis la fonction qui permet de passer de ces variétés aux images et à l'estimation de la posture 3D est apprise séparément. Ainsi, cette méthode permet de déterminer le point de vue et de reconstruire l'image en même temps que d'estimer la posture. Ceci permet de détecter d'éventuelles estimations abberrantes. Les auteurs de [Gall et al., 2010] proposent une approche qui se sert de plusieurs variétés de faibles dimensions qui sont apprises conjointement aux estimations de postures incorporées aux variétés. Cette approche utilise de plus la reconnaissance d'action comme une distribution a priori pour un apprentissage basé sur un recuit simulé à base de particules [Van Laarhoven and Aarts, 1987]. Certaines méthodes cherchent à apprendre l'espace des postures pour contraindre l'espace des solutions : ceci peut être fait en plaçant les images similaires à proximité, pour éviter l'estimation directe de la position des articulations [Mori et al., 2015], alors que d'autres méthodes ont essayé de réduire la dimension de l'espace des postures pour faciliter leur estimation [Sminchisescu and Jepson, 2004].

3.3.4 Estimation de posture sur des images de profondeur

De nombreuses approches ont été proposées pour effectuer l'estimation de posture humaine sur des images de profondeur. Elles sont décrites dans cette partie. Nous ne présentons pas les approches focalisées sur l'estimation de la position de la main, dont une revue récente de l'état de l'art est décrite par [Li et al., 2019], ni les méthodes qui combinent l'utilisation de plusieurs types de données, telles que celles proposées par [Knoop et al., 2006] ou [Jain et al., 2011].

Les premiers travaux sur le sujet ont été proposés en 2001 par [Grammalidis et al., 2001]. La méthode s'appuie sur un modèle qui est utilisé pour produire une image de profondeur de synthèse. L'écart entre l'image de profondeur de synthèse et l'image de profondeur originale est minimisé en déformant itérativement le modèle. Pour initialiser le modèle, les auteurs se servent un algorithme d'espérance maximisation qui permet d'identifier le bras, l'avant-bras et la main, ainsi que de positionner deux articulations (le poignet et le coude). Une autre approche basée sur un modèle a été introduite pour le suivi de posture du haut du corps par [Zhu and Fujimura, 2007, Zhu et al., 2008]. Cette méthode s'appuie sur la détection de caractéristique anatomiques et une initialisation d'un modèle de la tête, du cou et du torse par détection de ces parties. L'estimation de la position des membres supérieurs est ensuite effectuée par une première détection des bras et des avants bras puis une localisation de leurs articulations. Un contrôle de la posture estimée est réalisé à l'aide d'une boucle de cinématique inverse. Elle a ensuite été étendue par une approche de suivi permettant de corriger les erreurs à l'aide d'inférence bayésienne [Zhu and Fujimura, 2010].

Plusieurs méthodes proposent l'utilisation d'un algorithme d'optimisation *Iterative Clo-*

sest Point (ou ICP) [Besl and McKay, 1992, Chen and Medioni, 1992], pour l'estimation de posture. [Demirdjian et al., 2003] estime la position des parties du corps en ayant recours à ICP dans des nuages de points 3D générés à l'aide d'un système stéréo. La position de chaque partie du corps est estimée indépendamment et des contraintes cinématiques sont appliquées à l'aide d'un classificateur SVM entraîné sur des données de *motion capture*. [Grest et al., 2005] présente par la suite une approche d'optimisation utilisant ICP pour l'estimation de la posture 3D sur des images de profondeur, traitées comme des nuages de points. L'estimation de posture est réalisée en faisant correspondre un modèle 3D d'humain à celui représenté sur l'image de profondeur. Une approche similaire est proposée par [Droeschel and Behnke, 2011] pour aligner un modèle 3D sur une image de profondeur, alors que [Ganapathi et al., 2012] étend l'algorithme ICP en imposant la contrainte que le squelette doit se situer à l'intérieur du nuage de point.

Certaines approches ont recours à la distance géodésique, car cette distance se conserve lorsque la posture change. Ces méthodes distinguent les points primaires, situés au centre et aux extrémités du corps, tels que le centre du torse, la tête, les mains et les pieds, situés à des extrema de distance, et les points secondaires, tels que les coudes et les genoux. La position des points secondaires est estimée à partir de la position des points primaires. [Ganapathi et al., 2010] présente une approche pour le suivi de la posture humaine fondé sur un modèle discriminatif qui détecte les mains, la tête et les pieds à l'aide de la distance géodésique et de descripteurs locaux qui permettent de différencier les différentes parties du corps [Plagemann et al., 2010]. Les autres parties du corps sont positionnées à l'aide d'un modèle local appliqué itérativement et qui exploite la chaîne cinématique du squelette. Une méthode similaire est présentée par [Schwarz et al., 2011]. Afin de gérer les occultations, les auteurs utilisent le flux optique entre les images successives. Le contrôle de la posture estimée est à nouveau réalisé à l'aide d'une boucle de cinématique inverse.

Quelques méthodes utilisent la comparaison à des exemples par recherche dans une base de données pour estimer la posture. Par exemple, [Baak et al., 2013] se sert des points primaires détectés par [Plagemann et al., 2010] pour rechercher dans une base de données la posture qui y correspond. Une autre approche se base sur la comparaison d'une image de profondeur à une base de données d'images de profondeur de synthèse [Siddiqui and Medioni, 2010]. L'algorithme recherche la posture optimale en s'appuyant sur une méthode de Monte-Carlo par chaîne de Markov. Le processus est accéléré en proposant des positions candidates pour les position des main, des avant-bras et de la tête. Enfin, les auteurs de [Ye et al., 2011], en traitant l'image de profondeur comme un nuage de points, cherchent le nuage de points de plus proche dans une base de données, pour compléter et raffiner la posture associé afin de fournir une estimation de posture finale.

Enfin, certaines approches utilisent des méthodes probabilistes pour estimer la posture. [Holt et al., 2011] propose les *Connected Poselets* pour l'estimation de posture du haut du corps. Les *poselets*, extraits de données d'entraînement, sont tout d'abord détectés

sur l'image de profondeur. Une forêt aléatoire d'arbres de décision est ensuite utilisée pour reconstruire la posture complète de l'utilisateur. Un modèle graphique est ensuite mis en place pour modéliser les relations entre les différentes parties du corps et extraire la posture la plus probable. Une méthode décrite par [Wei et al., 2012] présente une estimation de la posture réalisée à l'aide d'un estimateur de *Maximum A Posteriori*, et l'estimation de posture est itérativement affinée grâce à des solveurs linéaires, avec des a priori sur la silhouette et la géométrie du corps. Une détection initiale de la posture permet d'automatiser un système de suivi. [Ye and Yang, 2014] introduisent un algorithme pour l'estimation conjointe de posture et de forme pour les objets articulés, tels que les animaux et les humains. L'algorithme proposé se sert d'un modèle de mélange gaussien, où la posture est estimée à l'aide d'une estimation de la densité de probabilité d'un modèle de déformation articulé et paramétré. Ainsi, il ne nécessite aucune mise en correspondance explicite entre les points, et est moins sensible aux minima locaux et aux mouvements rapides et complexes.

Les méthodes ayant eu le plus grand retentissement sont cependant celles proposées par l'équipe ayant travaillé sur l'estimateur de posture du capteur Kinect, développé par Microsoft. Suite à la commercialisation du capteur, la plupart des travaux requérant une estimation de posture 3D fiable sur des images de profondeur utilisent l'algorithme directement intégré dans le kit de développement du capteur, et plusieurs études ont permis de mesurer sa précision et son efficacité, en environnement contrôlé [Wang et al., 2015, Plantard et al., 2015], pour l'analyse de l'activité de personnes agées [Obdržálek et al., 2012], mais aussi en tant qu'outil d'évaluation de l'ergonomie [Plantard et al., 2017a, Plantard et al., 2017b]. Dans l'approche proposée par [Shotton et al., 2011] l'estimation de la position 3D des articulations est d'abord vue comme un problème de reconnaissance d'objets. En effet, le corps humain est divisé en sous-parties, et l'objectif est alors de réaliser la classification de chaque pixel du corps, pour en reconstruire les parties. Finalement la position pixellique des articulations est estimée en calculant le mode de différentes combinaisons de sous-parties du corps par une approche similaire à *mean shift* [Comaniciu and Meer, 2002, Zheng et al., 2009], et la troisième coordonnée est simplement égale à la valeur de profondeur du pixel à laquelle un ajoute un léger offset. Pour l'apprentissage nécessaire à l'étape de classification, les auteurs ont recours à des forêts d'arbres de décision qui sont entraînées sur un ensemble de données de synthèse. Les caractéristiques sont construites en calculant la différence entre les valeurs de profondeur normalisées de paires de pixels voisins du pixel étudié, elles sont donc simples et peu coûteuses à calculer. Cette approche a ensuite été améliorée dans [Girshick et al., 2011a], où les auteurs proposent une approche qui se passe de l'étape de classification des pixels : une forêt de regression permet d'attribuer un poids à chaque pixel pour chaque articulation, et la position finale des articulations est déterminée après un vote de chaque pixel, puis par [Sun et al., 2012], où les auteurs montrent qu'ajouter de l'information, tel que des relations de dépendance entre

les positions des articulations ou la taille de l'utilisateur permet d'améliorer la précision de l'estimation de posture.

L'estimation de posture proposée par la Kinect a par ailleurs été utilisée comme a priori d'autres méthodes, qui cherchent à la raffiner pour augmenter sa précision. Par exemple, on peut citer les travaux de [Shum et al., 2013], qui évalue la précision d'une posture prédite par le capteur Kinect puis l'optimise à l'aide d'une recherche des plus proches voisins dans une base de données de postures. De même, les approches basées sur les arbres de régression ont été approfondies en utilisant la marche aléatoire [Yub Jung et al., 2015], où les feuilles des arbres de décision proposent une direction à suivre pour trouver la position des articulations, ou bien par l'utilisation d'arbres binaires de décision (aussi appelés fougères, en anglais *ferns*) [Hesse et al., 2015].

3.4 Apprentissage par réseaux de neurones et apprentissage profond

Depuis 2012 et la victoire du réseau de neurones AlexNet [Krizhevsky et al., 2012] au concours de classification d'images ImageNet, les réseaux de neurones artificiels ont pris une part importante dans le paysage de la recherche en vision par ordinateur. Avec l'amélioration de la puissance de calcul des cartes graphiques et la disponibilité accrue de grandes masses de données, leur efficacité et leur large domaine d'application en font des outils prisés par les chercheurs et les industriels. Cette partie présente un bref historique des réseaux de neurones ainsi que les différents types d'architecture utilisés au cours de cette thèse.

3.4.1 Bref historique des réseaux de neurones

Si l'essor des réseaux de neurones pour la résolution de problèmes complexes est relativement récent, le concept de réseaux de neurones artificiels a été introduit en 1943 : dans cet article, Warren McCulloch et Walter Pitts exposent une théorie selon laquelle l'unité de base de l'activité cérébrale est l'activation des neurones [McCulloch and Pitts, 1943]. Ils définissent ainsi le neurone formel, qui est vu comme une fonction qui transforme des valeurs reçues en entrée en une valeur de sortie. En pratique, pour obtenir cette sortie, le neurone calcule la somme de ses entrées pondérées par des coefficients appelés poids synaptiques puis applique une fonction d'activation à seuil. Le neurone formel, illustré en Figure 3.18, est le premier modèle mathématique du neurone biologique et est un élément de base des réseaux de neurones artificiels.

La difficulté est de construire ces fonctions. L'apprentissage d'un réseau de neurones revient donc à optimiser les poids synaptiques de chaque neurone afin de résoudre au

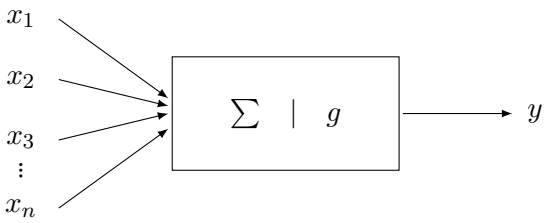


FIGURE 3.18 – La réprésentation d'un neurone formel. Un neurone formel prend en entrée une liste de valeurs (les x_i), applique une somme pondérée puis une fonction d'activation g pour calculer sa sortie y.

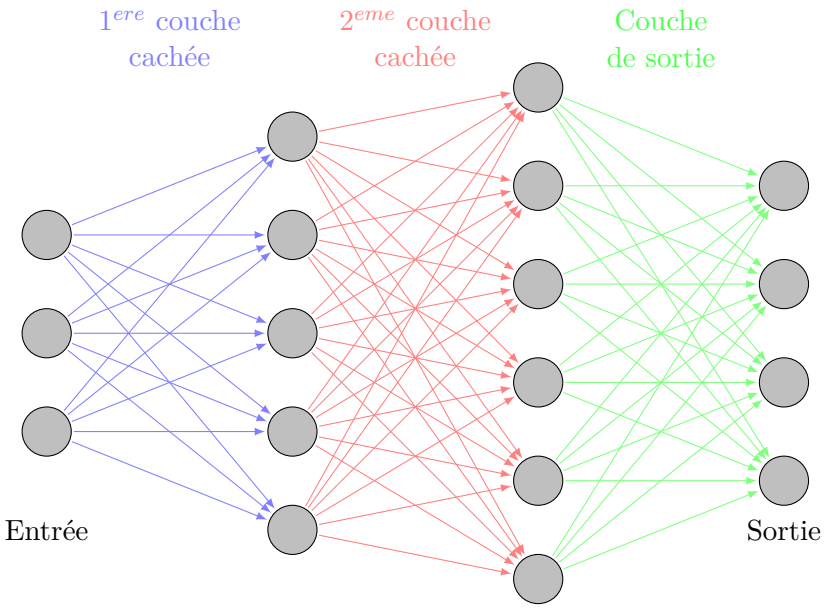


FIGURE 3.19 – Exemple de perceptron multicouche à trois couches. Le réseau de neurones contient deux couches cachées et une couche de sortie. Il prend en entrée un vecteur de taille trois. La première couche cachée est de taille cinq, la seconde de taille six. La sortie est un vecteur de taille quatre.

mieux le problème posé.

Donald Hebb [Hebb, 1949] présente par la suite la "règle de Hebb", qui donne un moyen de modifier la valeur de ces paramètres : si deux neurones qui se suivent sont activés en même temps, alors la force de leur connexion augmente. Cette règle est toujours présente dans le processus d'apprentissage des réseaux de neurones, où le poids synaptique d'une connexion se renforce lorsque les deux neurones liés par cette connexion s'activent en même temps. Cette loi est souvent vue comme un des principes de bases de l'apprentissage non supervisé [Baldi, 2012].

Le premier perceptron (mono-couche) a été proposé par Frank Rosenblatt [Rosenblatt, 1958] en 1958. C'est un classifieur linéaire binaire : il est capable de donner une classe à un vecteur numérique donné en entrée. Cependant, en 1969, Marvin Lee Minsky et Seymour

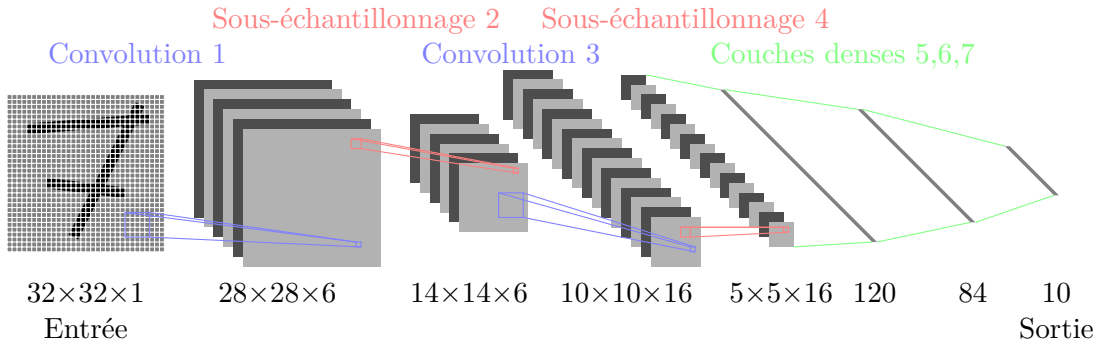


FIGURE 3.20 – L'architecture du réseau LeNet5 [LeCun et al., 1998]. L'entrée est une image en niveaux de gris de taille 32×32 et la sortie un vecteur de taille 10 représentant la probabilité d'appartenance à chacune des 10 classes. Ce réseau est composé de cinq couches (on ne compte pas les sous-échantillonnages).

Papert montrèrent les limites du perceptron et des classifieurs linéaires en général [Minsky and Papert, 1969]. En effet, les perceptrons ne sont capables de résoudre que des problèmes linéairement séparables.

Les perceptrons multicouches sont proposés par [Rumelhart et al., 1986]. Capables d'être entraînés et de résoudre des problèmes non-linéaires, ils permettent ainsi de résoudre de nouvelles catégories de problèmes qui n'étaient pas solvables à l'époque. Ils sont basés sur des travaux de Paul Werbos en 1974 [Werbos, 1974] puis de Yann Le Cun [Le Cun, 1986] et David Rumelhart [Rumelhart et al., 1986] en 1986, qui présentent la méthode de rétropropagation du gradient, pour permettre l'apprentissage sur des réseaux multi-couches d'un point de vue pratique. La rétropropagation redistribue l'erreur à travers les couches et permet de modifier les poids de chaque neurone. Cependant, l'apprentissage des réseaux de neurones est très lent et l'augmentation de la profondeur du réseau, c'est-à-dire de son nombre de couches, rend la convergence vers une solution de plus en plus difficile, voire impossible [Erhan et al., 2009]. Un exemple illustratif de perceptron multicouche est donné en Figure 3.19.

Les réseaux de neurones convolutifs font leur apparition au cours des années 1990. Ces réseaux de neurones, spécialement construits pour le traitement d'image, appliquent des filtres de convolution successifs dont les paramètres sont appris pendant l'entraînement du réseau de neurones. Un des travaux marquants de cette période est le réseau de neurones LeNet5 [LeCun et al., 1998], conçu pour la reconnaissance de caractères manuscrits et utilisé dans l'industrie par AT&T pour la lecture automatique de chèques (voir Figure 3.20 pour une description de l'architecture détaillée du réseau de neurones).

Mais c'est suite à l'augmentation de la puissance de calcul des cartes graphiques et du calcul distribué que les réseaux que les réseaux de neurones se popularisent au début des années 2010 : en 2010, Dan Claudiu Cireşan [Cireşan et al., 2010] présente une des

premières implémentations de réseaux de neurones faisant appel aux cartes graphiques dans le cadre de la classification de chiffres du dataset MNIST (Mixed National Institute of Standards and Technology) ; en 2012, Alex Krizhevsky propose le réseau de neurones AlexNet [Krizhevsky et al., 2012] qui gagne largement la compétition ImageNet [Deng et al., 2009], un concours de classification d'image à grande échelle (plusieurs millions d'images réparties en plusieurs milliers de classes), réduisant l'erreur de classification de 26% à 15%. Cette victoire aux "jeux olympiques annuels de la vision par ordinateur" met ainsi les réseaux de neurones sous la lumière des projecteurs.

C'est à partir de là que le terme apprentissage profond (en anglais *deep learning*) s'est popularisé pour qualifier l'utilisation de réseaux de neurones contenant de plus en plus de couches et devenant donc de plus en plus profonds et complexes.

Depuis, les réseaux de neurones se sont imposés comme des méthodes performantes pour la résolution d'une multitude de problèmes de différentes natures comme par exemple la classification [Simonyan and Zisserman, 2015, Szegedy et al., 2017], la segmentation [Ronneberger et al., 2015, Badrinarayanan et al., 2017], l'intelligence artificielle de jeux vidéos [Silver et al., 2016, Vinyals et al., 2019], et le traitement du langage naturel [Dahl et al., 2011, Graves and Jaitly, 2014].

3.4.2 Les différents types de réseaux utilisés dans cette thèse

Si les réseaux de neurones ont tant de succès, c'est parce qu'il en existe plusieurs types, adaptés à différentes situations. Cette partie présente uniquement les types de réseaux qui ont été utilisés dans le cadre des travaux et ne prétend pas être exhaustive. Elle se concentre en particulier uniquement sur des modèles utilisés en traitement d'images. Le livre *Deep Learning* [Goodfellow et al., 2016] de Ian Goodfellow, Yoshua Bengio et Aaron Courville fournit un panorama plus complet d'autres architectures, comme par exemple les réseaux de neuones récurrents largements utilisés en traitement du son et du langage naturel.

Réseaux de neurones convolutifs

L'architecture la plus simple est celle du perceptron multicouche [Rumelhart et al., 1986], déjà présenté précedemment (voir Figure 3.19). Les réseaux de neurones sont organisés en couches successives et les neurones d'une couche sont connectés à tous les neurones de la couche suivante, formant ainsi des couches dites denses ou complètement connectées (en anglais, *fully connected layers*).

Les réseaux adaptés au traitement d'image sont les réseaux de neurones convolutifs. LeNet5 [LeCun et al., 1998] (voir Figure 3.20), AlexNet [Krizhevsky et al., 2012] ou VGG [Simonyan and Zisserman, 2015] en sont des exemples. La première partie du réseau consiste en une partie convolutive, composée de couches de convolution (en anglais,

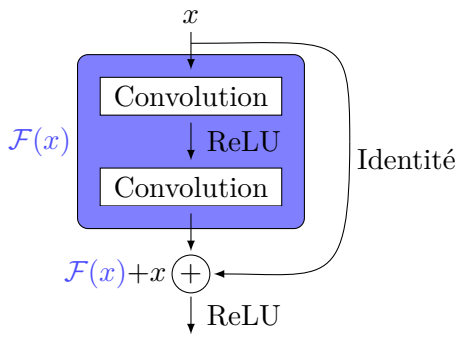


FIGURE 3.21 – Exemple de structure d'un bloc résiduel, composant de base des réseaux de neurones de type ResNet. Un bloc résiduel est constitué d'une série de couches de convolution. Entre ces couches, on a une activation de type *ReLU* (unité de rectification linéaire). On additionne la sortie des couches de convolution à l'entrée du bloc. Finalement, on applique une seconde activation *ReLU*.

convolutional layers) qui appliquent des filtres de convolution aux images puis aux cartes de caractéristiques successivement calculées, associées à des couches de "mise en commun" (en anglais, *pooling layers*), ayant pour but de réduire la taille spatiale des cartes de caractéristiques tout en conservant l'information obtenue par les couches de convolution. Les couches de *pooling* les plus utilisées sont les couches d'*average-pooling* et de *max-pooling* [Ranzato et al., 2007]. La seconde partie du réseau est un perceptron multicouche où se succèdent plusieurs couches denses. Cette seconde partie est facultative et souvent absente lorsque la sortie attendue du réseau est une image, comme par exemple dans [Long et al., 2015], qui présente un réseau de neurone pour la segmentation ne contenant qu'une partie convolutive.

Réseaux de neurones résiduels [He et al., 2016]

Si le nombre de couches d'un réseau de neurones peut théoriquement être infini, [He et al., 2016] constate que plus les réseaux de neurones sont profonds, plus ils sont difficiles à entraîner. En effet, cette difficulté, déjà exposée et expliqué par Sepp Hochreiter en 1998 [Hochreiter, 1998], peut venir de la dissipation du gradient (en anglais *vanishing gradient*) lors de l'étape de rétropropagation : multipliée par de petits coefficients, la valeur du gradient diminue au fur et à mesure qu'il remonte vers les premières couches du réseau de neurones. Cependant, les auteurs de [He et al., 2016] montrent qu'une augmentation de la profondeur provoque une saturation des performances puis un dégradation rapide des résultats, qui n'est pas directement liée à la dissipation du gradient, mais plutôt à un problème d'optimisation : il est très difficile de converger vers l'identité pour une ou plusieurs couches d'un réseau de neurones. Ils proposent donc une nouvelle forme d'architecture de réseaux de neurones, basée sur des blocs résiduels (voir Figure 3.21). Le réseau de neu-

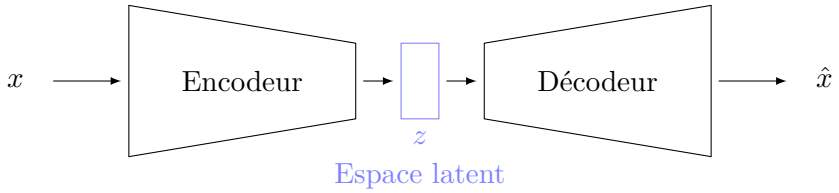


FIGURE 3.22 – Le principe de fonctionnement des auto-encodeurs. La donnée x est encodée sous la forme z qui évolue dans l'espace latent. Le décodeur décode ensuite l'information z pour reconstruire la donnée $\hat{x}$.

rones est constitué d'un enchaînement de blocs résiduels. Ces blocs sont composés d'une succession de couches convolutives, à laquelle on applique une connexion par saut [Bishop et al., 1995], c'est-à-dire que l'on additionne l'entrée du bloc à sa sortie. Les auteurs montrent alors que remplacer les couches convolutives par des blocs résiduels permet de construire des réseaux de neurones très profonds dont les performances s'améliorent avec l'augmentation de la profondeur.

Les réseaux de neurones résiduels ont été étendus par les réseaux de neurones convolutifs densément connectés (DenseNet) [Huang et al., 2017] : ce type de réseau est composé de blocs de couches convolutives où la sortie de chaque couche est concaténée à la sortie de toutes les couches suivantes. L'entrée d'une couche est donc composée de toutes les sorties des couches qui la précèdent à l'intérieur du bloc. Cette architecture favorise la réutilisation des caractéristiques précedemment apprises, et nécessite un nombre de paramètres réduit par rapport à ses excellentes performances.

D'autres réseaux de neurones, sans être des réseaux de neurones résiduels, utilisent des connexions par saut dans leur architecture : c'est le cas du U-net [Ronneberger et al., 2015], un réseau de neurone qui doit son nom à sa forme en U qui connecte la dernière couche convolutive de chaque niveau de la phase descendante à la première couche convolutive de la phase montante située au même niveau.

Auto-encodeurs [Kramer, 1991]

Les auto-encodeurs sont des réseaux de neurones très utilisés pour l'apprentissage non supervisé. Ils suivent le même principe que les algorithmes de compression : ils utilisent un ensemble de caractéristiques de dimension plus faible que celle d'entrée pour représenter l'information originale. Cette information compressée évolue dans un espace latent, qui est ensuite décodé pour reconstruire la donnée d'entrée. Les auto-encodeurs sont donc divisés en deux parties distinctes : un encodeur, qui encode l'information à partir des données d'entrée, et un décodeur, qui décode l'information stockée dans l'espace latent pour reconstruire les données (voir Figure 3.22).

Les auto-encodeurs ont donc ouvert la voie aux modèles génératifs pour l'apprentissage non supervisé, capables de générer des données en suivant une loi de probabilité apprise

grâce à des données fournies pendant l'apprentissage du réseau. En effet, pris séparement, le décodeur est un modèle génératif : il est capable de générer une image à partir d'information latente. Un exemple d'utilisation est l'application FakeApp [4], une application permettant de remplacer un visage par un autres sur des images. Cette application utilise deux auto-encodeurs entraînés chacun sur des images contenant le visage d'une personne : pour effectuer le changement de visage, il suffit d'intervertir les décodeurs.

Une variante de l'auto-encodeur est le modèle encodeur-décodeur, où le décodeur n'a plus pour objectif de reconstruire l'image données en entrée, mais une transformation de cette image grâce aux données compressées dans l'espace latent. Un exemple d'application est la segmentation sémantique avec SegNet [Badrinarayanan et al., 2017].

Réseaux antagonistes génératifs [Goodfellow et al., 2014]

Les réseaux antagonistes génératifs (en anglais *Generative adversarial network* et abrégé GAN) proposés en 2014 par Ian Goodfellow [Goodfellow et al., 2014, Goodfellow, 2016] permettent la mise en place de modèles permettant de générer efficacement des données avec un fort degré de réalisme. Dans ce modèle, deux réseaux sont placés en compétition :

— un générateur qui a pour mission de générer des échantillons ;
— un discriminateur qui a pour but de différencier les vrais échantillons des échantillons générés.

Ces deux réseaux apprennent de manière concurrente : le générateur se sert des résultats obtenus par le discriminateur pour construire des échantillons de plus en plus réalistes, tandis que le discriminateur affine sa capacité de détection. Une analogie peut être celle du faussaire et du policier, où le faussaire cherche à construire de la fausse monnaie de plus en plus réaliste, tandis que le policier détecte des faux de plus en plus perfectionnés. Une illustration du fonctionnement des GANs est donnée en Figure 3.23.

Cette situation est celle d'un jeu à deux joueurs à somme nulle. On sait donc qu'il existe un équilibre de Nash, c'est-à-dire une stratégie pour le générateur et le discriminateur où chacun minimise le gain maximal de l'adversaire [Goodfellow et al., 2014]. C'est la solution optimale du problème.

L'entraînement de ces réseaux de neurones est cependant dificile en pratique [Salimans et al., 2016]. Plusieurs écueils peuvent se présenter au cours de l'entraînement :

— **non-convergence** : les paramètres du modèle oscillent et ne convergent jamais ;
— **effondrement** (en anglais *mode collapse*) : le générateur s'effondre et ne produit plus qu'une quantité limitée d'échantillons ;
— **diminution du gradient** : si le discriminateur est trop efficace, le gradient du générateur disparaît et celui-ci ne peut plus apprendre ;
— **sensibilité aux choix des hyper-paramètres** : la sélection des hyper-paramètres est difficile et dépend du problème à résoudre.

4. https://www.malavida.com/en/soft/fakeapp

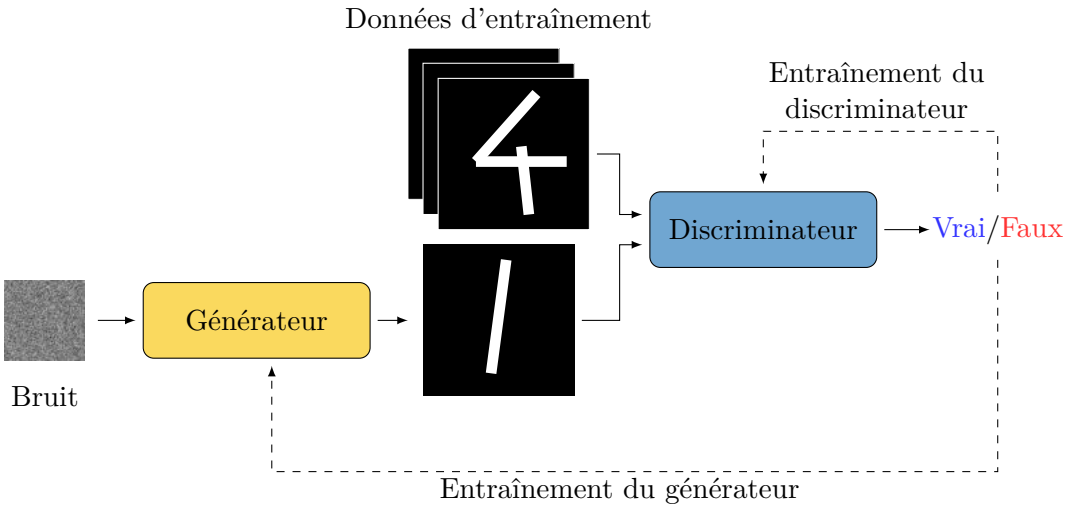


FIGURE 3.23 – Exemple de fonctionnement des réseaux antagonistes génératifs pour la génération d'images de chiffres. Le générateur construit des images réalistes à partir de bruit donné en entrée. Le discriminateur sépare les données d'entraînement des données fabriquées par le générateur. La rétropropagation suite aux résultats du discriminateur permet d'entraîner les deux réseaux de manière antagoniste.

Les GAN ont été étendus par [Mirza and Osindero, 2014] : en ajoutant une information supplémentaire de condition en entrée, ils rendent le générateur capable de générer des données conditionnées par des étiquettes de classe.

De nombreux types de GAN ont ensuite été présentés comme les réseaux de neurones antagonistes convolutifs [Radford et al., 2015] (abrégé DCGAN pour *Deep Convolutional Generative Adversarial Network*), qui sont une première tentative de réseaux génératifs antagonistes utilisant des couches convolutives et donc spécialisés dans la génération d'images réalistes ou les réseaux de neurones antagonistes à classifieur auxiliaire [Odena et al., 2017] (abrégé ACGAN pour *Auxiliary Classifier Generative Adversarial Network*), où le discriminateur prédit la classe des données qu'il reçoit en entrée en plus de donner une discrimination entre les données réelles et les données générées.

Les GANs ont obtenu des résultats impressionnants en matière d'édition d'images [Wu et al., 2019, Zhu et al., 2016, Perarnau et al., 2016], génération d'images [Denton et al., 2015, Radford et al., 2015] et de génération conditionnelle [Reed et al., 2016, Zhang et al., 2017a]. Les GANs sont également utilisés dans d'autres domaines comme les données 3D [Wu et al., 2016, Wang et al., 2017] ou les vidéos [Vondrick et al., 2016, Tulyakov et al., 2018].

Nom	Équation	Dérivée	Étendue	Problème associé
ReLU	$f(x) = \begin{cases} 0 \text{ si } x < 0 \\ x \text{ sinon} \end{cases}$	$f'(x) = \begin{cases} 0 \text{ si } x \leq 0 \\ 1 \text{ sinon} \end{cases}$	$[0, +\infty[$	—
LeakyReLU	$f(x) = \begin{cases} \alpha x \text{ si } x < 0 \\ x \text{ sinon} \end{cases}$	$f'(x) = \begin{cases} \alpha \text{ si } x < 0 \\ 1 \text{ sinon} \end{cases}$	$[-\infty, +\infty[$	—
Sigmoïde	$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$[0, 1]$	Classification binaire
Softmax	$f(v)_i = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}$	$\frac{\partial f_i(v)}{\partial x_j} = f_i(v)(\delta_{ij} - fj(v))$	$[0, 1]$	Classification multi-classe
Tangente hyperbolique	$f(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$	$[-1, 1]$	Génération d'images
Linéaire	$f(x) = x$	$f'(x) = 1$	$]-\infty, +\infty[$	Régression

TABLE 3.1 – Description des principales fonctions d'activation et types de problèmes associés.

3.4.3 Paramétrage des réseaux de neurones

Si l'architecture d'un réseau de neurones est déterminante afin que le modèle soit capable de résoudre au mieux un problème donné, d'autres paramètres sont tout aussi importants pour optimiser les capacités d'un modèle ainsi que son apprentissage. En particulier, on aborde ici les différentes fonctions d'activation et leur pertinence en fonction du problème à résoudre, de même que les fonctions de pertes qui peuvent être utilisées. Ensuite, nous définissons les différents ensembles de données que l'on peut retrouver lors de l'entraînement d'un réseau de neurones. Enfin, nous décrivons le principe du *boosting*, qui permet de combiner plusieurs apprenants dits "faibles" afin d'optimiser les résultats du modèle global.

Fonctions d'activation

Des composants indispensables au bon fonctionnement des réseaux de neurones sont ses fonctions d'activation. Ce sont des fonctions mathématiques, généralement non linéaires, qui sont appliquées en sortie d'une couche de convolution ou d'une couche dense, servant à reproduire le "potentiel d'activation" biologique du cerveau. La non linéarité des fonctions d'activation permet de faire du réseau de neurones un approximateur de fonctions universel [Cybenko, 1989]. Les fonctions d'activations sont la plupart du temps dérivables partout, pour pouvoir effectuer une descente de gradient, monotones et avec un point fixe en 0. Dans cette partie, nous présentons les fonctions d'activation utilisées dans les travaux présentés dans cette thèse.

Ces fonctions d'activation sont résumées dans le Tableau 3.1.

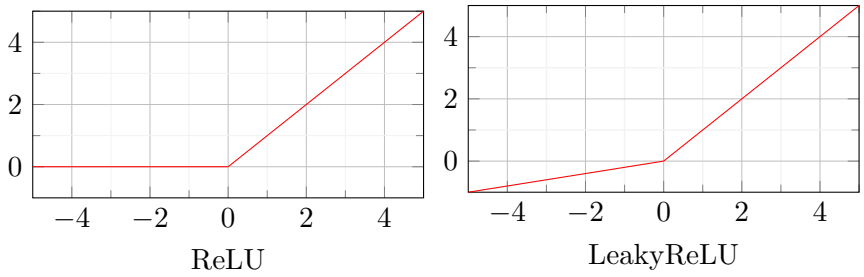


FIGURE 3.24 – Les fonctions d'activation ReLU et LeakyReLU (ici, $\alpha = 0,2$). La différence entre ces deux fonctions se situe au niveau des valeurs négatives : pour tout $x < 0$, on a $ReLU(x) = 0$ et $LeakyReLU(x) = \alpha x$. Ainsi, le gradient de la fonction ReLU pour $x < 0$ est nul, alors que celui de la fonction LeakyReLU est égal à α.

Unité de rectification linéaire (*Rectified Linear Unit*, ou ReLU)

La fonction d'activation ReLU (de l'anglais *Rectified Linear Unit*) [Glorot et al., 2011], est une des fonctions d'activation les plus couramment utilisées dans l'apprentissage profond. Elle s'exprime de la manière suivante :

$$ReLU(x) = \begin{cases} 0 \text{ si } x < 0 \\ x \text{ sinon} \end{cases} \tag{3.2}$$

Pour toutes les valeurs supérieures à zéro, elle se comporte comme une fonction linéaire, et sa non linéarité vient du faire qu'elle donne zéro pour tout antécédent négatif. Elle est intéressante car elle est peu couteuse en calcul (c'est simplement la fonction $max(0, x)$), mais elle a le défaut de mettre toutes les valeurs négatives à zéro, ce qui peut poser problème appelé *dying ReLU problem*, où le neurone concerné donne toujours zéro et n'est plus en mesure d'apprendre, car le gradient vaut aussi zéro lors de la rétropropagation.

Une tentative de corriger ce problème est la fonction d'activation LeakyReLU. Proposée par [Maas et al., 2013], elle s'exprime de la manière suivante :

$$LeakyReLU(x) = \begin{cases} \alpha x \text{ si } x < 0 \\ x \text{ sinon} \end{cases} \tag{3.3}$$

Ceci permet donc d'éviter l'écueil du *dying ReLU problem*, car le gradient des valeurs négatives est maintenant égal à α.

Les courbes des fonctions d'activation ReLU et LeakyReLU sont présentées en Figure 3.24.

Sigmoïde et softmax

La fonction sigmoïde, aussi appelée logistique, est une fonction d'activation à valeur dans $[0, 1]$, utilisée pour exprimer la probabilité d'une épreuve de Bernouilli, c'est-à-dire une épreuve pour laquelle il n'y a que deux sorties possibles. Elle est donc généralement utilisée comme fonction d'activation de la dernière couche d'un réseau de neurones de classification binaire, tel que le discriminateur d'un modèle de réseaux antagonistes génératifs. Pour tout $x \in \mathbb{R}$, la fonction sigmoïde est définie par :

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{3.4}$$

La fonction softmax, aussi appelée fonction exponentielle normalisée, est une généralisation de la fonction sigmoïde, utilisée pour representée une loi catégorielle. Elle est donc utilisée comme fonction d'activation de la dernière couche d'un réseau de neurones de classification multi-classes, c'est-à-dire dont le nombre de classes finales est strictement supérieur à deux. La fonction softmax prend en entrée un vecteur $v = (x_1, x_2, \ldots, x_k) \in \mathbb{R}^k$, et est définie par :

$$\sigma(v)_i = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \text{ pour tout } i \in [\![1, k]\!] \tag{3.5}$$

Tangente hyperbolique

La fonction tangente hyperbolique est utilisée lorsqu'on souhaite obtenir une sortie dans l'intervalle $[-1, 1]$. Ainsi, on s'en sert souvent comme fonction d'activation de la dernière couche du générateur d'un modèle de réseaux antagonistes génératifs, pour obtenir en sortie une image normalisée entre et [-1,1]. Pour tout $x \in \mathbb{R}$, la fonction tangente hyperbolique est définie de la manière suivante :

$$\tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{3.6}$$

Activation linéaire

La fonction d'activation linéaire est en général utilisée en sortie d'un réseau de neurones utilisée pour un problème de régression. Les modèles qui l'utilisent omettent souvent de la mentionner, car son action est imperceptible. En effet, elle ne modifie pas la valeur qui lui est donnée en entrée, et son gradient est toujours égal à 1 ; pour tout $x \in \mathbb{R}$, elle est définie par :

$$lineaire(x) = x \tag{3.7}$$

Entraînement des réseaux de neurones

Les performances d'un réseau de neurones, si elles dépendent fortement de l'architecture de celui-ci, dépendent aussi directement de son entraînement. Nous décrivons dans

cette section les principaux paramètres qui influent sur l'entraînement d'un modèle d'apprentissage profond, mais qui peuvent être étudié pour l'apprentissage automatique en général.

Séparation des données pour l'entraînement

Lors de l'entraînement d'un réseau de neurones, on divise souvent notre ensemble de données en trois catégories :

— l'**ensemble d'apprentissage** ou **ensemble d'entraînement**, constitué des **données d'apprentissage** ou **données d'entraînement**, qui servent à entraîner le modèle : le modèle utilise ces données pour optimiser ses paramètres ;

— l'**ensemble de validation**, qui regroupe les **données de validation**, utilisées pour valider l'entraînement du modèle : ces données ne servent pas à entraîner le modèle mais seulement à optimiser les hyper-paramètres d'apprentissage et à vérifier que l'apprentissage se déroule correctement en testant le modèle régulièrement pendant sa phase d'entraînement ;

— l'**ensemble de test** ou **ensemble d'évaluation**, composé des **données de test** ou **données d'évaluation**, qui permettent de mesurer la capacité de généralisation du modèle après l'apprentissage : elle servent à simuler l'utilisation du modèle sur de nouvelles données.

Chacun de ces ensembles doit être assez représentif de l'ensemble des données qui seront amenées à être présentées au modèle. Souvent, on simplifie la découpe en utilisant les mêmes données pour la validation et pour le test. Cela introduit cependant un biais, car les données de test ont alors une influence sur l'apprentissage. Dans l'idéal, les données de validation et de test doivent être différentes, et les données de test ne doivent être utilisées que pour la phase de test du modèle [Hastie et al., 2009].

Époque (en anglais *epoch*) et itération

En apprentissage profond, une époque est définie comme un cycle d'entraînement sur l'ensemble des données d'entraînement. La phase d'apprentissage dure généralement plusieurs époques, c'est-à-dire que le modèle voit plusieurs fois les mêmes données d'entraînement. Il faut différencier l'époque de l'itération. Une itération est une étape élémentaire d'apprentissage où le réseau de neurones effectue une propagation vers l'avant puis une rétropropagation sur un petit lot (en anglais *batch*) de données. Très souvent, une époque dure donc plusieurs itérations : par exemple, si on a mille données qu'on divise en lots de cent données, une époque dure dix itérations.

Fonctions de perte (en anglais *loss functions*)

Une fonction de perte permet de mesurer la différence entre les valeurs réelles attendues, notées y, et les valeurs calculées par un modèle, notées $\widehat{y}$. La perte correspond à une

pénalité de mauvaise prédiction. Pendant la phase d'entraînement du réseau de neurones, on cherche donc à minimiser cette fonction de perte, en modifiant itérativement les poids du réseau de neurones. On appelle ce procédé minimisation du risque empirique.

Plusieurs fonctions de perte peuvent être utilisées en fonction de la tâche à réaliser par le modèle. Nous présentons ici les deux fonctions de perte auxquelles nous avons eu recours pour entraîner les modèles développés au cours de la thèse :

— l'**entropie croisée** (en anglais *cross-entropy*, abrégé CE), qui est utilisée pour les problèmes de classification ou de segmentation, et donc pour mesurer des différences de probabilité. Pour N classes ou régions, elle est définie par :

$$CE(y, \widehat{y}) = -\sum_{i=1}^{N} y_i \log \widehat{y}_i \tag{3.8}$$

Dans le cas particulier de la segmentation et de la classification binaires, c'est-à-dire lorsqu'il n'y a que deux classes ou deux régions à segmenter, l'**entropie binaire croisée** (en anglais *binary cross-entropy*, abrégé BCE), s'exprime par :

$$BCE(y, \widehat{y}) = -y \log \widehat{y} - (1 - y) \log(1 - \widehat{y}) \tag{3.9}$$

— l'**erreur quadratique moyenne** (en anglais *mean squared error*, abrégé MSE), est utilisée pour les problèmes de régression. Elle est définie par :

$$MSE(y, \widehat{y}) = \frac{1}{n} \sum_{i=1}^{n} |y_i - \widehat{y}_i|^2 \tag{3.10}$$

Optimiseurs (en anglais *optimizers*)

Si les fonctions de pertes permettent de quantifier les erreurs de prédiction du modèle, il reste à manipuler ces erreurs pour modifier les paramètres du modèles pour minimiser ces fonctions. C'est le rôle des optimiseurs : ils relient la fonction de perte et les paramètres du modèles en mettant à jour celui-ci par rapport aux écarts de prédiction mesurés, et ce sont donc les optimiseurs qui permettent d'optimiser le modèle. Ces optimiseurs possèdent en particulier un paramètre, appelé **taux d'apprentissage**, qui est un coefficient multiplicateur permettant de fixer la vitesse de mise à jour des paramètres par l'optimiseur. Un taux d'apprentissage trop faible entraîne un temps d'apprentissage trop long, alors qu'un taux d'apprentissage trop élevé entraîne des oscillations autour du minimum, qui peut donc ne jamais être atteint. Il est aussi possible de tenir compte du **momentum** de mise à jour, c'est-à-dire conserver une information sur la direction de mise à jour des paramètres du modèles des itérations précédentes, afin que le modèle converge plus rapidement.

L'optimiseur le plus populaire est la descente de gradient, dite stochastique si une époque dure plusieurs itérations. Le principe de cet algorithme est d'avancer itérativement dans le sens inverse du gradient, afin de réduire la valeur de la fonction de perte.

Un autre optimiseur utilisé dans cette thèse est l'optimiseur Adam (dérivé de l'anglais *adaptive moment estimation*, traduisible par estimation adaptative du moment) [Kingma and Ba, 2015], qui adapte le taux d'apprentissage à l'aide d'estimations de la moyenne et de la variance non centrée du gradient.

Boosting

En apprentissage automatique, le *boosting* est un méta-algorithme destiné à optimiser les performances d'un ensemble de classifieurs. Le principe est de combiner des "apprenants faibles" pour en faire un "apprenant fort" final. Les apprenants faibles sont des algorithmes avec des performances faiblement supérieures à celles du hasard.

Des exemples de *boosting* sont AdaBoost (de l'anglais *adaptive boosting*) et l'agrégation bootstrap (abrégé *bagging* pour *bootstrap aggregating*).

Adaboost [Freund and Schapire, 1995] combine la sortie de plusieurs apprenants faibles à l'aide d'une somme pondérée pour effectuer la prédiction finale. C'est un procédé adaptatif car les apprenants faibles sont modifiés à l'aide des mauvaises réponses données par les apprenants faibles précédents. AdaBoost est cependant sensible aux données bruitées ainsi qu'aux valeurs aberrantes.

3.5 Qualité d'un apprentissage

Différents phénomènes et mesures permettent de mesurer la qualité d'un apprentissage, en particulier lorsqu'on utilise des modèles d'apprentissage profond. Ils sont présentés dans cette partie.

Ces phénomènes et mesures sont réalisés en découpant nos jeux de données en trois ensembles distincts décrits en Section 3.4.3.

Pour illustrer les phénomènes proposés, on s'appuie sur l'ensemble de données et un apprentissage basé sur ces données considéré comme correct présentés en Figure 3.25.

3.5.1 Mesures utilisées pour évaluer l'apprentissage et le modèle

Pour effectuer l'apprentissage et pour évaluer sa qualité, on utilise des mesures qui permettent de calculer un écart entre la réponse donnée par le modèle et la réponse attendue lorsque l'on fournit une certaine donnée en entrée du modèle. Ces différentes mesures sont illustrées en Figure 3.26.

Une première mesure utilisée est l'**erreur empirique**, ou erreur d'entraînement : c'est l'écart moyenne mesurée entre la prédiction du modèle et la sortie attendue pour les **données d'entraînement**. Cette erreur est celle qui est utilisée pour améliorer le modèle pendant l'apprentissage, car c'est celle-ci que le modèle va chercher à minimiser pendant l'entraînement.

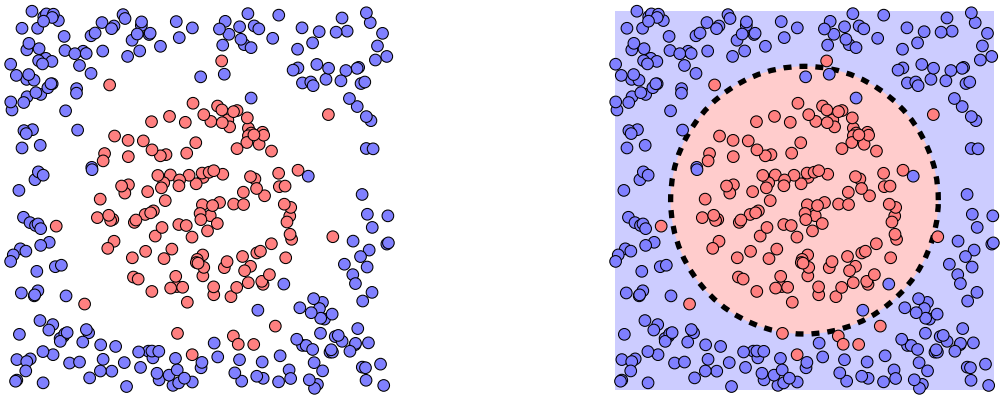


FIGURE 3.25 – Exemple de données appartenant à deux classes (à gauche, en bleu et rouge) et de résultat d'un apprentissage pour la classification de ces données en deux classes (à droite) : les données appartenant au domaine en bleu sont classées dans la classe bleue, tandis que les données appartenant au domaine rouge sont classées dans la classe rouge. On observe que les données ne sont pas parfaitement classées par le modèle : certaines données bruitées n'appartiennent pas au bon domaine. Le modèle semble cependant avoir bien analysé la structure des données.

Une seconde mesure utilisée est l'**erreur de généralisation**, ou erreur de validation : c'est l'erreur moyenne mesurée entre la prédiction du modèle et la sortie attendue pour les **données de validation**. Cette erreur permet d'estimer la capacité de généralisation du modèle. En effet, elle est mesurée sur des données que le modèle n'utilise pas pour s'entraîner, et correspond donc à une estimation de l'erreur du modèle lorsqu'on lui présente de nouvelles données.

L'écart entre ces deux mesures est appelé **écart de généralisation** : c'est la différence entre les performances du modèle sur les données d'entraînement par rapport à ses performances sur de nouvelles données [Jiang et al., 2018].

3.5.2 Sous-apprentissage

Le sous-apprentissage, aussi appelé sous-ajustement (en anglais *undertraining* ou *underfitting*), est un problème qui peut survenir dans le cadre de l'apprentissage supervisé. On dit qu'un modèle est dans une situation de sous-apprentissage lorsque qu'il ne parvient pas à capturer correctement la structure sous-jacente des données, et ne peut donc pas être utilisé.

Le sous-apprentissage apparaît souvent lorsque le modèle utilisé est trop simple pour résoudre le problème posé, ou qu'il n'est pas assez entraîné : ce modèle sous-adapté manque de paramètres ou ceux-ci ne sont pas correctement configurés [Everitt and Skrondal, 2006]. Un exemple de modèle sous-appris est présenté en Figure 3.27, à gauche.

Le sous-apprentissage se repère lorsque l'erreur empirique semble être anormalement éle-

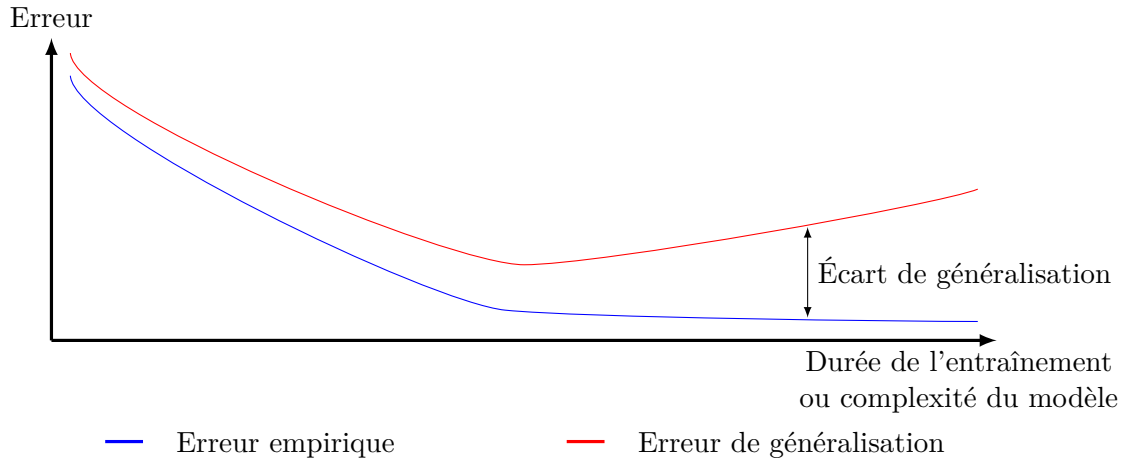


FIGURE 3.26 – Les différentes mesures utilisées pour analyser les performances d'un modèle d'apprentissage supervisé : l'erreur empirique (en bleu) et l'erreur de généralisation (en rouge). L'écart entre ces deux erreurs est l'écart de généralisation.

vée. L'entraînement d'un modèle démarre en général par une phase de sous-apprentissage (voir Figure 3.28), en particulier lorsqu'il est entraîné à partir de zéro.

Pour corriger les problèmes de sous-apprentissage, il y a deux leviers principaux. Premièrement, on peut augmenter la complexité du modèle, pour que celui-ci ait la capacité de représenter correctement la structure des données. Deuxièmement, on peut augmenter la durée de l'entraînement, afin que le modèle ait plus de temps pour adapter ses paramètres aux données.

3.5.3 Surapprentissage

Le surapprentissage, aussi appelé sur-ajustement (en anglais *overtraining* ou *overfitting*), est le problème inverse du sous-apprentissage. On parle de surapprentissage lorsque le modèle correspond trop précisément à un ensemble particulier de données, en l'occurence les données utilisées pendant l'entraînement, et qu'il peut donc ne pas correspondre à de nouvelles données.

En général, le surapprentissage survient lorsque le modèle utilisé pour résoudre une tâche est trop complexe, c'est-à-dire qu'il contient plus de paramètres que le problème n'en nécessite [Everitt and Skrondal, 2006], et qu'il est entraîné pendant trop longtemps. Ainsi, le modèle va parfaitement s'adapter aux données, en extrayant une partie du bruit contenu dans les données comme si celui-ci était représentatif des données [Burnham and Anderson, 2002] : le modèle devient donc trop sensible aux données utilisées pendant l'entraînement. Le modèle a alors perdu sa capacité de généralisation, et ne sera pas fiable lorsqu'il sera utilisé sur de nouvelles données. Un exemple de modèle ayant surappris est

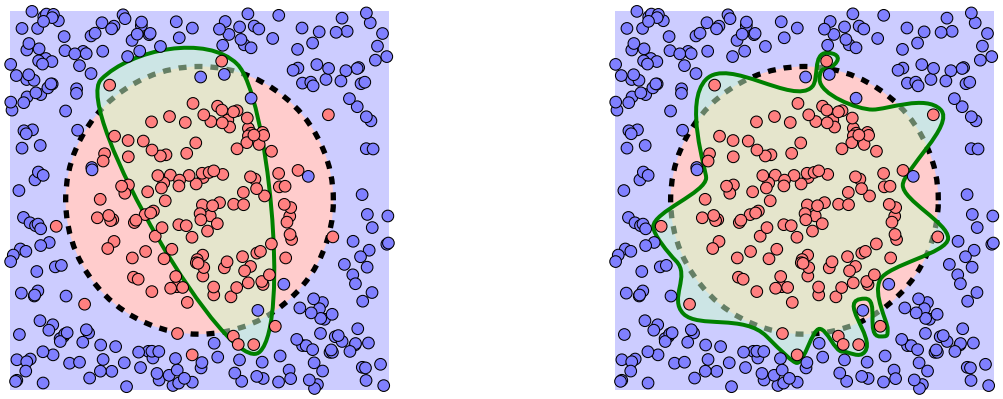


FIGURE 3.27 – Exemple de sous-apprentissage (à gauche) et de surapprentissage (à droite). À gauche, la courbe séparatrice, en vert, est trop simple pour diviser correctement les deux ensembles de données d'entraînement : le modèle n'est pas adapté aux données. À droite, la courbe séparatrice, en vert, divise parfaitement les deux ensembles de données d'entraînement, mais est très dépendante des données. Elle perd donc sa capacité de généralisation, en donnant trop d'importance à des observations bruitées.

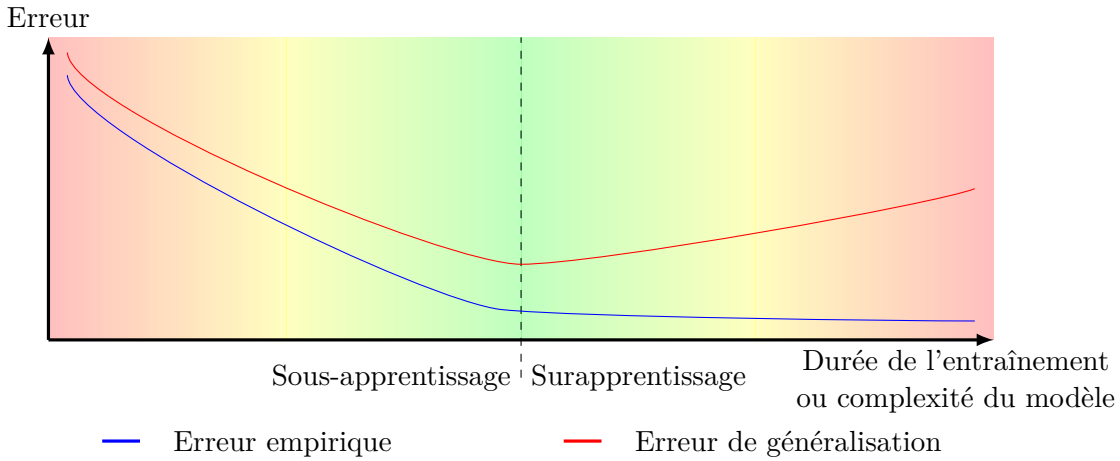


FIGURE 3.28 – Mesure du sous-apprentissage et du surapprentissage à l'aide de l'erreur empirique (en bleu) et de l'erreur de généralisation (en rouge). Au début de l'apprentissage, ou pour un modèle trop simple, on a un sous-apprentissage : l'erreur empirique est élevée et l'erreur de généralisation n'a pas encore atteint de minimum. En revanche, l'apprentissage prolongé et une complexité trop importante du modèle peuvent provoquer un sur-apprentissage. Celui-ci continue de s'améliorer sur les données d'entraînement, mais perd sa capacité de généralisation.

donné en Figure 3.27, à droite.

Le surapprentissage peut être repéré à l'aide de l'écart de généralisation et de l'erreur de généralisation (voir Figure 3.28) : lorsque l'écart entre l'erreur empirique et l'erreur de validation devient important, il est probable que le modèle soit dans une situation de surapprentissage.

Pour éviter le surapprentissage d'un modèle, on peut recourir à plusieurs procédés :

— ajouter de la régularisation : par exemple, en utilisant plus de données ou en utilisant un terme de régularisation pour l'optimisation des paramètres ;
— réduire la complexité du modèle : en réduisant le nombre de paramètres, le modèle ne peut plus tenir compte du bruit ;
— arrêter l'entraînement plus tôt : lorqu'on observe qu'on a atteint un écart de généralisation minimum, on peut décider d'arrêter l'apprentissage.

En résumé

Dans ce chapitre, nous présentons un état de l'art des différentes méthodes utilisées pour la segmentation d'image et l'estimation de posture humaine, qui constituent les deux principales parties du processus d'analyse de la pénibilité développé pendant cette thèse. Nous introduisons par ailleurs l'apprentissage profond et les réseaux de neurones, qui ont été utilisés dans les travaux réalisés au cours de la thèse.

Tout d'abord, nous nous intéressons en Section 3.1 à l'apprentissage automatique en général, en nous focalisant sur l'apprentissage supervisé et non supervisé, qui sont les deux principaux types d'apprentissage utilisés par les approches de l'état de l'art.

Nous présentons ensuite les principales méthodes "classiques" pour la segmentation d'image couleur (voir Section 3.2). Nous abordons les approches de segmentation basées sur le seuillage et le regroupement, la découpe en régions et l'utilisation de contour, puis les méthodes utilisant l'apprentissage. Nous terminons cette partie par une description de la principale méthode pour la segmentation d'images de profondeur.

Nous proposons en Section 3.3 un état de l'art des principales méthodes traditionnelles pour l'estimation de posture humaine. Nous présentons ainsi les approches basées sur des modèles définis a priori, puis celles fondées sur l'utilisation de données contenant un couple image/posture, dont font partie les méthodes d'estimation de posture par apprentissage. Comme pour la segmentation, cette partie s'achève par une description des méthodes d'estimation de posture sur les images de profondeur.

Nous introduisons par la suite l'apprentissage profond et les réseaux de neurones (voir Section 3.4). Après un bref historique, nous décrivons les principaux types de réseaux de neurones utilisés dans l'état de l'art et au cours de la thèse. Nous présentons les principaux paramètres d'un modèle d'apprentissage profond, ainsi que les outils nécessaires à leur entraînement.

Enfin, nous terminons ce chapitre par la présentation des principaux phénomènes et mesures utilisés pour évaluer la qualité d'un apprentissage (voir Section 3.5).

Chapitre 4

Segmentation d'images de profondeur

Sommaire

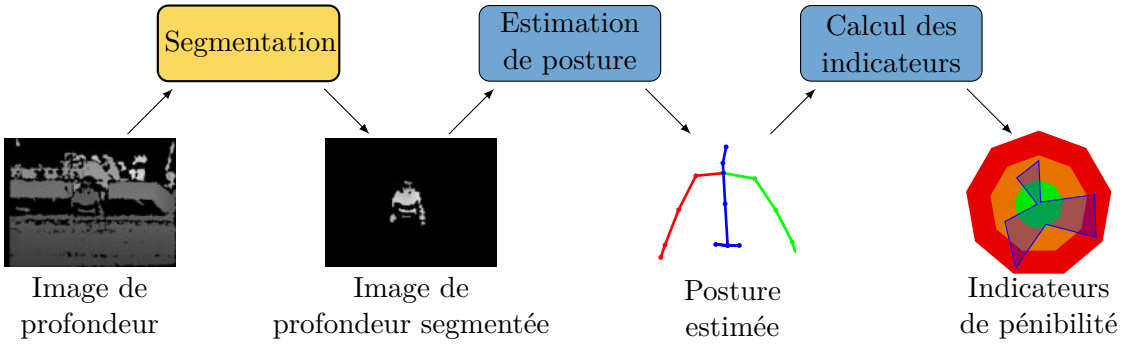


FIGURE 4.1 – Notre pipeline pour le calcul des indicateurs de pénibilité à partir des images de profondeur. Dans ce chapitre, nous construisons son premier module (en jaune), qui nous permet d'extraire l'opérateur humain du reste de l'image de profondeur.

Dans ce chapitre, nous cherchons à construire le premier module de notre pipeline, présenté en Figure 4.1. Nous souhaitons donc élaborer un outil nous permettant de réaliser, de manière automatisée, la segmentation d'images de profondeur, afin d'extraire l'opérateur humain du reste de l'image, avant d'estimer sa posture (voir Chapitre 6). Ainsi, nous proposons une approche pour la segmentation d'images de profondeur par apprentissage faiblement supervisé, illustré en Figure 4.2.

Tout d'abord, nous effectuons une segmentation automatique (voir Section 4.4). Cette segmentation automatisée nécessite pour seule interaction la sélection d'une zone par l'utilisateur. À ce moment-là, la qualité de la segmentation est très hétérogène, et s'avère insuffisante pour servir de base à l'estimation de la posture de l'opérateur.

Nous cherchons ensuite à utiliser un réseau de neurones pour améliorer cette première segmentation imparfaite (voir Section 4.5). Ainsi, nous introduisons un processus d'apprentissage qui tient compte de la nature imparfaite des données, en filtrant itérativement l'ensemble de données pour ne conserver que les images les mieux segmentées.

Une série d'expériences, présentée en Section 4.6, montre que cette méthode améliore significativement la performance du réseau de neurones, et donc la qualité de la segmentation finale, aussi bien par rapport à un apprentissage réalisé sur l'ensemble des données d'entraînement que par rapport à la première segmentation automatique.

4.1 Rappel du contexte et des motivations

D'énormes progrès ont été réalisés durant les dernières années dans le domaine du traitement d'images, amenés par les avancées concernant l'apprentissage profond et l'accroissement de la puissance de calcul des processeurs graphiques. Des réseaux de neurones convolutifs de plus en plus profonds peuvent maintenant être entraînés grâce aux architectures basées sur les blocs résiduels et permettent de résoudre des problèmes d'une très grande

complexité. Cependant, toutes ces avancées nécessitent des quantités importantes de données d'apprentissage, qui sont coûteuses et peuvent dans certains cas être impossibles à obtenir. Un changement commence donc à apparaître, et les travaux s'orientent peu à peu vers des paradigmes d'apprentissage nécessitant moins de données, comme l'apprentissage non supervisé, faiblement supervisé ou en "quelques coups" (*few-shots learning*).

Le problème d'analyse de l'activité humaine en milieu industriel dans le but de prévenir l'apparition de pathologies telles que les troubles musculo-squelettiques, abordé dans cette thèse, demande le recours à des algorithmes d'apprentissage nécessitant peu ou pas de données annotées, car le contexte industriel implique que l'environnement, non-contrôlé, peut fortement varier d'un centre de tri à l'autre (voir Section 1.4). De plus, nous utilisons des capteurs de profondeur afin de préserver autant que possible la vie privée des opérateurs, et favoriser son acceptabilité par les travailleurs (voir Section 1.5). On rappelle que les capteurs sont placés au niveau du plafond, avec une vue en plongée sur l'opérateur monitoré, ce qui n'est pas le point de vue habituel des capteurs de profondeur, généralement utilisés pour du divertissement [Shotton et al., 2011], où l'utilisateur se trouve face à la caméra.

Toutes ces conditions difficiles créent des images de profondeur très bruitées dans lesquelles l'opérateur humain est difficile à détecter et à suivre, et nous ne pouvons pas nous fier aux approches existantes pour segmenter l'opérateur (voir Figure 4.3).

Étant donnés nos objectifs, nous nous concentrons sur le problème de la segmentation du corps humain dans des images de profondeur bruitées, afin d'effectuer ultérieurement des traitements de plus haut niveau, tels que la reconnaissance d'action [Chen et al., 2016a] ou l'estimation de la posture [Shotton et al., 2011]. Certains capteurs, comme la caméra Kinect de Microsoft, peuvent fournir une segmentation de l'utilisateur à partir de l'image de profondeur, mais ce n'est pas le cas pour la majorité des capteurs actuels. De plus, la plupart des jeux de données, comme celui de Microsoft Research Action3D [Li et al., 2010], fournissent une image déjà segmentée de l'utilisateur. En effet, un grand nombre de travaux actuels sur les images de profondeur exigent que l'utilisateur soit segmenté et extrait de la scène [Chen et al., 2016a, Shotton et al., 2011, Li et al., 2010], alors que le reste de l'image peut être considéré comme du bruit et pollue considérablement les résultats de la grande majorité des algorithmes, rendant souvent ceux-ci inutilisables en raison de leur manque de robustesse. Segmenter l'utilisateur du reste de l'image est donc une tâche cruciale lorsqu'il s'agit du pré-traitement d'images de profondeur. Dans le cadre de cette thèse, la segmentation de l'opérateur humain dans l'image est réalisée avec l'objectif d'estimer sa posture à une étape ultérieure.

Le contexte particulier de notre projet nous oblige à supposer que nous n'avons pas assez de données annotées pour construire un ensemble consistant de données d'entraînement, en raison de la variabilité de nos images : l'environnement peut varier et la morphologie et les vêtements des opérateurs humains sont également imprévisibles, demandant donc d'ef-

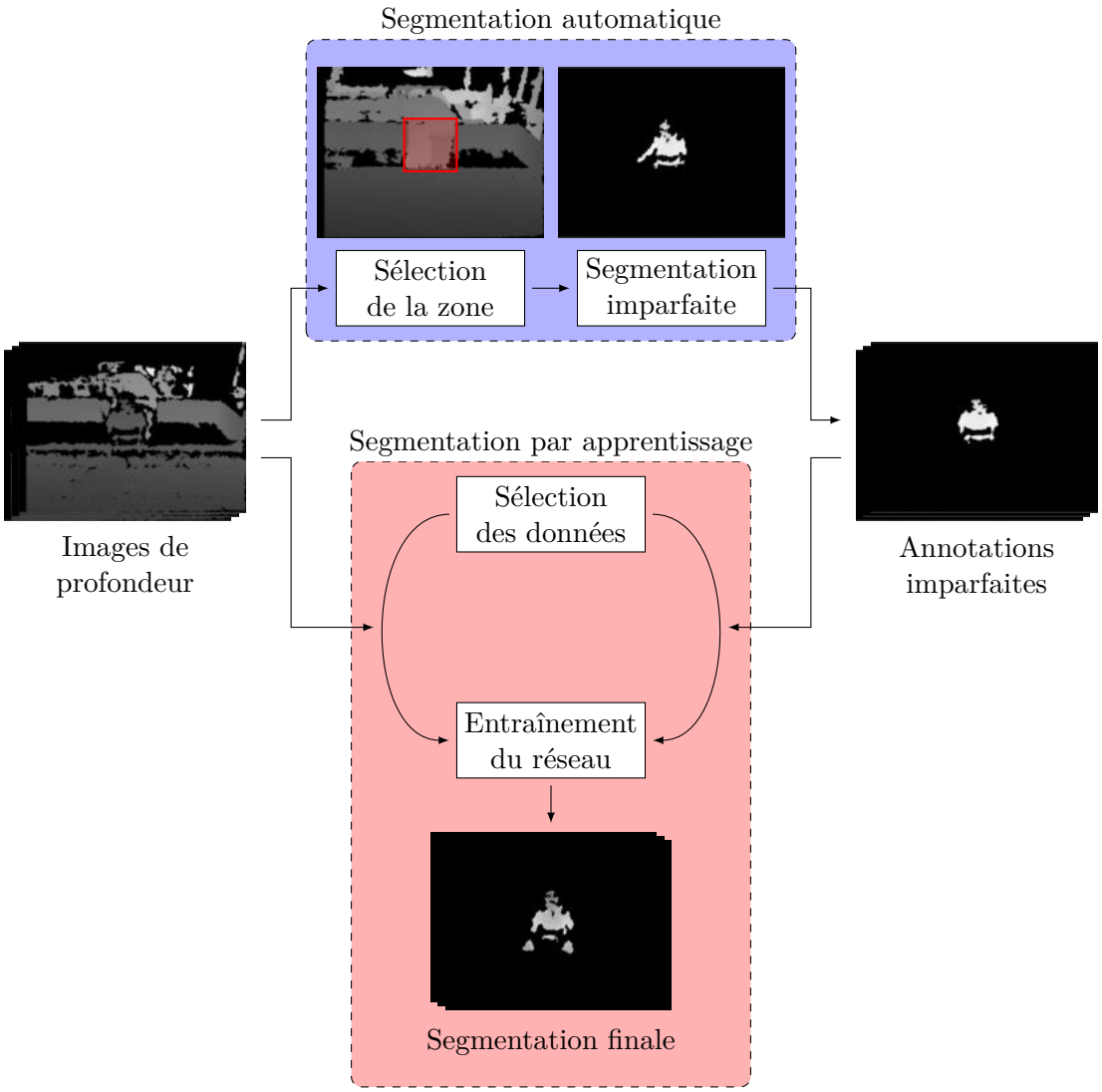


FIGURE 4.2 – Le déroulement du processus de segmentation proposé. La seule partie qui nécessite une interaction de l'utilisateur est la sélection de la zone avant d'effectuer la segmentation automatique. Le reste du processus est automatisé.

fectuer de nouvelles annotations, ce qui devient rapidement trop coûteux. Au lieu de cela, nous introduisons un processus (voir Figure 4.2) pour construire et affiner un ensemble de données annotées automatiquement, grâce à des techniques de traitement d'images de base et avec une faible supervision humaine.

FIGURE 4.3 – Exemple d'images de notre jeu de données (ligne du haut) et de leur segmentation de vérité terrain associée (ligne du bas). La segmentation de l'opérateur humain sur ces images est difficile en raison des contraintes de l'environnement industriel et de la nature bruitée des images de profondeur.

4.2 Apprentissage avec peu de données ou des données imparfaites

Dans cette partie, nous présentons différents types d'apprentissage qui permettent d'effectuer un apprentissage supervisé avec peu de données, ou avec des données de qualité insuffisante. Les méthodes présentées ici s'appuient souvent sur une approche d'apprentissage supervisé.

On ne traite pas ici de l'apprentissage semi-supervisé, qui est un ensemble d'approches permettant d'effectuer un apprentissage à l'aide de données annotées et de données non-annotées. On se concentre ici sur des méthodes qui utilisent des données annotées, disponibles en faible quantité, ou dont la qualité n'est pas optimale.

4.2.1 Apprentissage actif

L'apprentissage actif est une approche utilisée lorsque l'on possède un ensemble de données dont seule une petite partie est annotée. Cette approche suppose qu'il existe un "oracle", en général un expert humain, qui peut être interrogé dans le but d'obtenir des annotations pour certaines données non annotées.

On simplifie souvent le problème en estimant que le coût de l'annotation ne dépend que du nombre de requêtes d'annotation. L'objectif de l'apprentissage actif est donc de minimiser le nombre de demandes afin que le coût soit réduit au minimum. Étant donné le

petit nombre de données étiquetées, l'apprentissage actif tente de sélectionner les données les plus pertinentes à annoter.

Deux critères de sélection sont largement utilisés [Huang et al., 2010] :

— l'informativité, qui mesure combien un échantillon non étiqueté contribue à réduire l'incertitude du modèle ;

— la représentativité, qui mesure combien un échantillon représente la structure des données d'entraînement.

L'échantillonnage par incertitude [Lewis and Gale, 1994] (en anglais *uncertainty sampling*) et la requête par votes [Seung et al., 1992] (en anglais *query by commitee*) sont des approches basées sur l'informativité. L'échantillonnage par incertitude construit un modèle unique qui interroge l'oracle sur la donnée non annotée en laquelle il a le moins confiance. La requête par votes utilise plusieurs modèles et interroge l'oracle sur la donnée non annotée qui génère le plus de désaccord. Le défaut des approches basées sur la représentativité est qu'elles s'appuient beaucoup sur les données annotées pour construire le modèle initial qui permet de sélectionner les données à étiqueter, et les performances peuvent être instables si on ne dispose que de peu de données annotées.

Les approches fondées sur la représentativité cherchent à exploiter la structure des données non annotées, en utilisant par exemple des algorithmes de partitionnement [Nguyen and Smeulders, 2004, Dasgupta and Hsu, 2008]. Un inconvénient de ces méthodes est donc qu'elles dépendent beaucoup du résultat de l'algorithme de partitionnement.

Ainsi, des méthodes récentes tentent de combiner la réprésentativité et l'informativité [Huang et al., 2010, Wang and Ye, 2015].

4.2.2 Apprentissage faiblement supervisé

L'apprentissage faiblement supervisé (en anglais *weakly-supervised learning*) est une partie de l'apprentissage automatique où les données et les annotations utilisées dans le cadre d'un apprentissage supervisé viennent de sources bruitées ou imprécises. Cette approche diminue le coût représenté par la création de données annotées à la main. Ces annotations imparfaites sont utilisées en ayant la connaissance de leur imperfection, mais peuvent tout de même servir à construire des modèle solides [Zhou, 2017].

L'apprentissage faiblement supervisé permet donc de contourner des problèmes qui peuvent souvent survenir lorsque l'on souhaite utiliser des approches d'apprentissage automatique, comme une expertise insuffisante pour annoter les données de manière suffisamment précise, ou tout simplement un manque de temps.

Les annotations "faibles" peuvent prendre différentes formes [Zhou, 2017] :

— elles peuvent être incomplètes : sur un ensemble de données, seule une petite fraction est annotée. On se rapproche ici de la problématique de l'apprentissage semi-supervisé ou de l'apprentissage actif ;

— elles peuvent être inexactes : seules des annotations grossières sont données, par exemple une classe d'image plutôt qu'une segmentation des éléments de l'image ;
— elles peuvent être imprécises : il peut y avoir des erreurs et des imprécisions, par exemple si elles ont été acquises par *crowdsourcing*.

Les approches qui cherchent à résoudre des problèmes où les annotations sont incomplètes ont recours à l'apprentissage semi-supervisé (voir Section 6.2.1) ou à l'apprentissage actif (voir précédemment).

Les approches qui tentent d'effectuer des tâches où les annotations sont inexactes utilisent l'apprentissage multi-instance [Dietterich et al., 1997]. Ici, on traite les données par groupe, appelé sac : l'élément de base de l'apprentissage multi-instance est le sac de données [Zhou et al., 2009]. Les données qui contiennent la propriété recherchée sont dites "positives", les autres sont "négatives". Un sac de données est positif si au moins une des données du sac est positive, mais il peut donc aussi contenir des données négatives. L'apprentissage multi-instance peut avoir deux objectifs :

— inférer un concept qui permet d'étiqueter correctement les données individuelles ;
— apprendre à étiqueter les sacs sans inférer le concept.

Un exemple simple d'apprentissage multi-instance est imaginé par [Babenko, 2008] : on a une pièce fermée à clé, et plusieurs trousseaux (les "sacs") de clés (les "données"). Certains trousseaux permettent d'entrer dans la pièce, alors que d'autres non. La tâche consiste à prédire si une clé donnée ou un certain trousseau permet d'entrer dans la pièce. Pour résoudre le problème, on peut donc soit apprendre à trouver la clé qui permet d'ouvrir la porte, soit apprendre à classer les trousseaux entiers.

Enfin, les approches qui souhaitent résoudre des problèmes avec des données imprécises ont souvent pour objectif d'identifier puis de corriger les mauvaises annotations avant d'effectuer l'apprentissage [Brodley and Friedl, 1999]. Les cas suspects peuvent aussi être retirés des données d'entraînement.

4.2.3 Apprentissage par transfert

L'apprentissage par transfert (en anglais *transfer learning*) vise à utiliser un modèle entraîné pour résoudre un problème particulier, appelé problème source, puis à réutiliser ce modèle comme point de départ d'un modèle cherchant à traiter un problème similaire, appelé problème cible, en ne le modifiant que superficiellement [Pan and Yang, 2009].

Cette approche est privilégiée lorsque construire un modèle de zéro s'avère trop coûteux en temps et/ou en calculs : l'apprentissage par transfert permet d'accélérer la construction du modèle cible.

On peut distinguer trois types d'apprentissage par transfert :

— le transfert inductif : le problème cible est différent, les données source et cible peuvent être différentes. En fonction de l'utilisation ou non de données annotées

Type d'apprentissage	Données	Tâches à réaliser
Apprentissage traditionnel	Peuvent être différentes	Identiques
Transfert inductif	Identiques	Différentes
Transfert transductif	Pas de données cible	Identiques
Transfert non supervisé	Indisponibles	Différentes

TABLE 4.1 – Tableau des différents types d'apprentissage par transfert.

source pour résoudre le problème cible, ce type d'apprentissage peut à nouveau être divisé en deux catégories :

- des données source annotées sont disponibles : on les utilise en cherchant à atteindre des performances élevées dans la résolution du nouveau problème. Cette approche peut être vue comme de l'apprentissage multi-tâche (voir plus bas) ;
- il n'y a pas de données source annotées : on est alors dans le cadre de l'apprentissage autodidacte (en anglais *self-taught learning*) [Raina et al., 2007], où le modèle ne peut pas utiliser les annotations du premier problème.

— le transfert transductif [Arnold et al., 2007] : le problème à résoudre est le même, mais on ne possède pas de données annotées pour résoudre le problème cible. On trouve dans ce type d'apprentissage par transfert l'adaptation de domaine [Bengio, 2012, Zhang et al., 2019] ;

— le transfert non supervisé : les problèmes source et cible sont liés, mais différents, et on ne dispose pas de données annotées. On se retrouve ici dans un cas d'apprentissage non supervisé (voir Section 3.1.2). On a alors recours à des méthodes de partitionnement [Dai et al., 2008] ou de réduction de dimension [Pan et al., 2010].

Ces trois types d'apprentissage par transfert sont résumés dans le Tableau 4.1.

L'apprentissage multi-tâche peut être vu comme une partie de l'apprentissage par transfert inductif. C'est une approche qui cherche à résoudre plusieurs problèmes connexes en parallèle. Le but de l'apprentissage multi-tâche est de découvrir des caractéristiques communes qui peuvent profiter à chaque problème : ce qui est découvert pour résoudre un problème peut aider à résoudre les autres en augmentant la capacité du modèle à généraliser [Caruana, 1997].

Les approches d'apprentissage actif, bien qu'intéressantes lorsque l'on ne possède que peu de données annotées et que l'on souhaite annoter les données les plus pertinentes, ne semblent cependant pas utilisables dans notre cas. En effet, la segmentation manuelle d'images de profondeur est une tâche très coûteuse et parfois imprécise, et la grande variabilité qu'implique le contexte industriel et l'utilisation d'un modèle d'apprentissage profond demanderait d'annoter une grande quantité de données. Les méthodes d'appren-

tissage par transfert, quant à elles, sont intéressantes lorsqu'on possède déjà un modèle de référence, entraîné sur des données similaires à celles utilisées. Dans notre cas, il n'existe pas d'ensemble de données proche des nôtres, ni de modèle de l'état de l'art pré-entraîné. Ainsi, nous ne pouvons pas utiliser une telle approche.

Il reste donc les approches d'apprentissage faiblement supervisé. Dans notre cas, nous utiliserons des données imprécises, c'est-à-dire pouvant contenir des erreurs et des imprécisions. Ainsi, notre objectif sera d'identifier les mauvaises annotations au cours de l'apprentissage, pour les retirer des données d'entraînement.

4.3 Segmentation d'image par réseaux de neurones

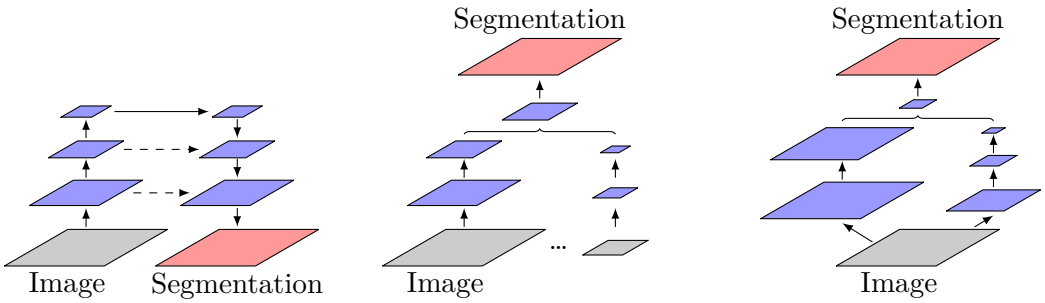
Cette partie se concentre maintenant sur les travaux ayant recours à l'apprentissage profond et aux réseaux de neurones pour la segmentation d'image. Comme pour les algorithmes d'apprentissage supervisé présentés précédemment (voir Section 3.2.4), les réseaux de neurones effectuent généralement une segmentation sémantique de l'image. De plus, ils se heurtent au même problème d'un besoin de prise en compte coinjointe de la localité de chaque pixel associée à une compréhension globale de l'image.

Alors que les approches d'apprentissage profond sont devenues populaires pour la classification d'images en 2012 [Krizhevsky et al., 2012], les premières méthodes de segmentation réutilisent ce principe en cherchant à classer les pixels de l'image : elles vont classer des patchs construits par fenêtre glissante sur l'image, pour finalement donner une classe au pixel central de chaque patch [Farabet et al., 2013, Cireşan et al., 2012].

Les travaux révolutionnaires en segmentation d'images sont arrivés plus tard avec les réseaux de neurones entièrement convolutifs [Long et al., 2015] et U-net [Ronneberger et al., 2015]. Les réseaux de neurones entièrement convolutifs sont des réseaux qui ne comportent que des blocs convolutifs, c'est-à-dire des couches de convolution associées à des couches d'activations, de régularisation et de mise en commun, mais pas de couche complètement connectée, à la différence par exemple de LeNet5 [LeCun et al., 1998], dont les trois dernières couches sont des couches denses. Ainsi, ils sont capables de traiter directement une image dans son ensemble.

Au cours de cette étude bibliographique, nous avons distingué trois types d'approches pour résoudre cette problématique, avec une prise en compte conjointe de la localité et du contexte :

— des approches basées sur les auto-encodeurs : la phase descendante (partie encodeur) permet d'obtenir une compréhension sémantique de l'image, puis une phase ascendante (partie décodeur) permet de retrouver une précision pixellique. Au cours de la phase ascendante, on réutilise souvant des caractéristiques construites pendant la



Approche encodeur-décodeur [Ronneberger et al., 2015] | Approche multi-résolution [Roy and Todorovic, 2016] | Approche parallèle [Yu et al., 2018]

FIGURE 4.4 – Illustration des différents types d'approches pour la segmentation sémantique basée sur les réseaux de neurones. Tous les réseaux d'une même approche n'ont pas la même architecture, mais partagent la même idée sur la manière de prendre en compte à la fois la localité et le contexte global.

phase descendante ;

— des approches fondées sur un traitement multi-résolution de l'image : on utilise un modèle pyramidal où l'image est donnée avec une résolution différent à chaque niveau. Deux approches se distinguent : l'information de chacun des niveaux peut être concaténée au niveau d'une dernière couche de classification, ou les étages de résolution supérieure peuvent utiliser les caractéristiques calculées à des résolution inférieure ;

— des approches où le calcul de l'information locale et globale est effectué en parallèle : le réseau est divisé en plusieurs branches dont l'information est à un certain moment rassemblée pour construire la segmentation finale.

Ces trois types d'approches sont illustrées en Figure 4.4.

Dans la suite de cette partie, nous exposons, pour chacune des trois types d'approche, des réseaux de neurones marquant de l'état de l'art.

4.3.1 Approches basées sur une architecture encodeur-décodeur

Une première approche pour résoudre le problème de la segmentation sémantique par réseaux de neurones consiste à réaliser une première étape descendante avec diminution progressive de la résolution pour obtenir une comprehension sémantique, puis d'effectuer une deuxième phase ascendante afin de redistribuer l'information au niveau des pixels et de retrouver la résolution initiale de l'image. Les différentes méthodes présentées ici se basent sur cette approche.

Jonathan Long est le premier à avoir proposé l'utilisation de réseaux complètement convolutifs pour la segmentation sémantique d'image [Long et al., 2015]. Il montre que les réseaux de neurones obtiennent de meilleurs résultats que les approches traditionnelles. Le

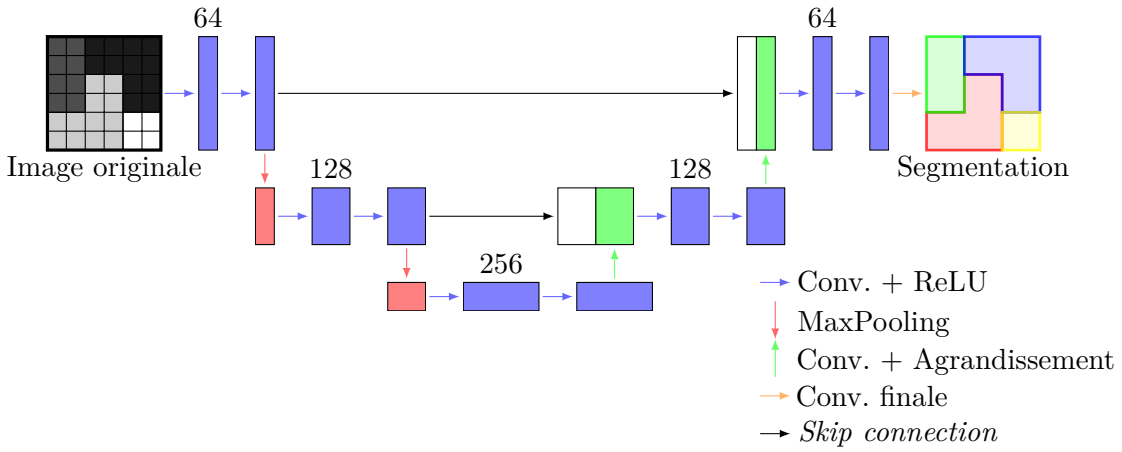


FIGURE 4.5 – Exemple d'architecture d'un réseau de neurones de type U-net. Les flèches bleues représentent les couches convolutives associées à des activations ReLU, les flèches rouges les couches de *max-pooling* et les flèches vertes des couches de convolution et de suréchantillonnage. Les flèches noires représentent les *skip connection*, les données concaténées sont affichées en blanc.

réseau de neurones proposé procède par convolution et réductions de dimension successives, puis par un suréchantillonnage progressif qui permet de retrouver une sortie aux mêmes dimensions que l'image originale.

U-net est un réseau de neurones entièrement convolutif proposé par Olaf Ronneberger en 2015 [Ronneberger et al., 2015], développé pour la segmentation d'images médicales. Ce réseau, inspiré d'une architecture d'auto-encodeur, est une amélioration du premier réseau poposé pour la segmentation sémantique [Long et al., 2015]. Son architecture caractéristique en forme de U, permet de connecter des couches peu profondes de la partie descendante à des couches profondes de la partie ascendante via des *skip connection*. Cette méthode permet ainsi, à l'issue de la phase descendante, d'obtenir une compréhension sémantique de l'image dans son ensemble, puis lors de la phase ascendant de retrouver une précision pixellique pour cette information sémantique [Ronneberger et al., 2015]. Un exemple d'architecture de U-net est donnée en Figure 4.5. Ce modèle a été étendu par U-net++ [Zhou et al., 2018], où les *skip connection* sont remplacées par des couches convolutives densément connectées [Huang et al., 2017].

Le réseau de neurones SegNet a été présenté en 2017 par Vijay Badrinarayanan [Badrinarayanan et al., 2017]. Inspiré d'une architecture d'encodeur-décodeur, ce réseau de neurones contient un encodeur construit en suivant l'architecture de la partie convolutive du réseau VGG-16 [Simonyan and Zisserman, 2015], et un décodeur construit en miroir à l'encodeur. La particularité de ce réseau est que les indices d'échantillonnage des valeurs utilisées dans les couches de *max-pooling* sont réutilisés dans les couches de suréchantillonnage. Ces couches construisent donc des matrices creuses, qui sont ensuite convoluées pour

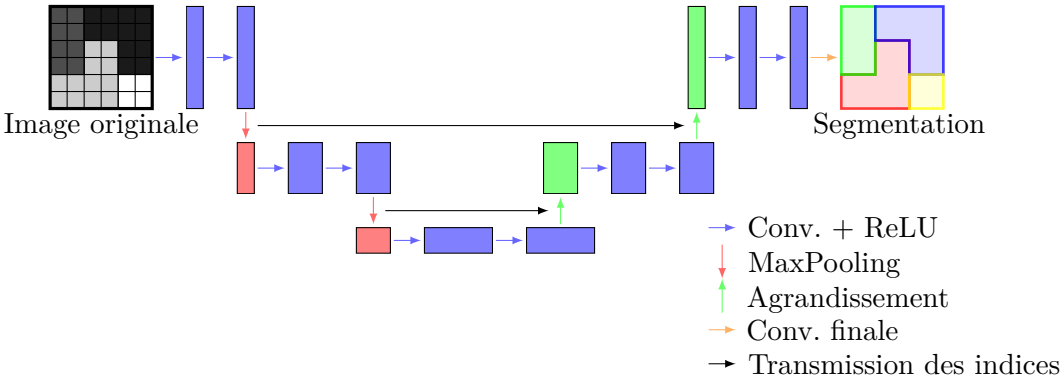


FIGURE 4.6 – Exemple d'architecture d'un réseau de neurones de type SegNet. Les flèches bleues représentent les couches convolutives associées à des activations ReLU, les flèches rouges les couches de *max-pooling* et les flèches vertes des opérations de suréchantillonnage qui suivent les indices transmis par les couches de *max-pooling* (flèches noires).

construire des cartes de caractéristiques et une segmentation de tous les pixels. Une architecture simplifiée du réseau SegNet est illustrée en Figure 4.6. Ce modèle a été étendu par le SegNet bayesien (*Bayesian SegNet*) [Kendall et al., 2017]. En plus de proposer une segmentation, le SegNet bayesien donne une carte d'incertitude qui expose les zones difficiles à segmenter, en général les frontières entre les différentes régions.

Pedro Pinheiro propose en 2015 le modèle DeepMask [Pinheiro et al., 2015]. L'approche est basée sur un unique réseau convolutif. Pour un patch d'entrée, le réseau propose à la fois un masque de segmentation et une classification du patch en fonction de l'objet qu'il contient. Les deux tâches sont apprises en parallèle par le réseau, qui se divise en deux branches pour les dernières couches, une branche construisant le masque et l'autre la classification. Les mêmes auteurs présentent ensuite SharpMask [Pinheiro et al., 2016]. Basé sur l'architecture DeepMask, SharpMask se sert des sorties des premières couches pour raffiner une première sortie en basse résolution du masque de segmentation. Le fonctionnement du réseau est le suivant : à la manière du U-net [Ronneberger et al., 2015], SharpMask utilise DeepMask comme partie descendante. La partie ascendante affine le masque proposé en utilisant les sorties des couches de la partie descendante.

4.3.2 Segmentation multi-résolution

Certaines approches proposent une segmentation multi-résolution. Ces méthodes utilisent des réseaux de neurones qui fonctionnent à différentes résolutions, c'est-à-dire que l'image donnée en entrée est redimensionnée, avec une résolution propre à chaque réseau. Dans [Eigen and Fergus, 2015], les sorties des réseaux travaillant à faible résolution sont utilisées, après redimensionnement, par les réseaux à une résolution plus importante. [Roy and Todorovic, 2016] propose plutôt une sortie commune à tous les réseaux, les images

de faible résolution étant redimensionnées pour retrouver la résolution originale. RefineNet [Lin et al., 2017] propose un modèle multi-résolution utilisant des blocs résiduels distincts pour chaque niveau de résolution. Ces résultats sont ensuite fusionnés grâce à une phase de redimensionnement, puis par une dernière phase de *chained résidual pooling*, ou plusieurs blocs composés d'une couche de *max-pooling* et d'une couche de convolution se succèdent dans l'objectif de tenir compte d'un large contexte.

D'autres méthodes proposent une segmentation multi-résolution réalisée à l'aide de réseaux de neurones récurrents. Dans [Pinheiro and Collobert, 2014], le même réseau est appliqué plusieurs fois en série. En partant de la résolution initiale, une première itération dans le réseau propose une segmentation à une résolution inférieure à laquelle est concaténée l'image redimensionnée. Les itérations se succèdent et le réseau propose finalement une classe pour chaque pixel. Basé sur le modèle ReNet [Visin et al., 2015], ReSeg [Visin et al., 2016] s'appuie sur une architecture VGG-16 [Simonyan and Zisserman, 2015] pour extraire des cartes de caractéristiques. Ces cartes de caractéristiques sont ensuite raffinées à l'aide de réseaux de neurones récurrents.

[Byeon et al., 2015] propose une méthode utilisant des réseaux de neurones récurrents contenant des blocs de mémoire à long et à court terme (en anglais *long short term memory*, abrégé LSTM) en deux dimensions. L'image est divisée en patchs qui ne se chevauchent pas. Ces patchs sont donnés en entrée de quatre blocs LSTM distincts. Cette approche a une complexité de calcul très faible.

4.3.3 Méthodes de traitement parallèle de la localité et du contexte

Un troisième type d'approches essaie de traiter la localité et le contexte en même temps, ou en parallèle. Ces approches utilisent généralement des architectures plus complexes, qui utilisent par exemple des convolutions "atrous" [Chen et al., 2015] ou un réseau divisé en deux branches travaillant en parallèle [Yu et al., 2018].

Les convolutions atrous (du français "à trous"), aussi appelées convolutions dilatées, c'est-à-dire des couches convolutives dont les filtres de convolution sont plus grands mais contiennent des trous réguliers (voir Figure 4.7) : ils ont ainsi un champ réceptif plus étendu et peuvent chercher de l'information plus lointaine, avec l'avantage d'avoir le même nombre de paramètres qu'un filtre de convolution traditionnel.

DeepLab est un modèle pour la segmentation sémantique d'image proposé en trois versions. Toutes ces versions sont fondées sur des convolutions "atrous", qui prennent à chaque itération un peu plus d'importance dans le processus de segmentation.

En 2015, Liang-Chieh Chen introduit un modèle baptisé DeepLab [Chen et al., 2015]. Ce modèle utilise un réseau de neurones couplé à des champs aléatoires conditionnels complétement connectés. Le réseau de neurones est construit de manière à transformer l'image

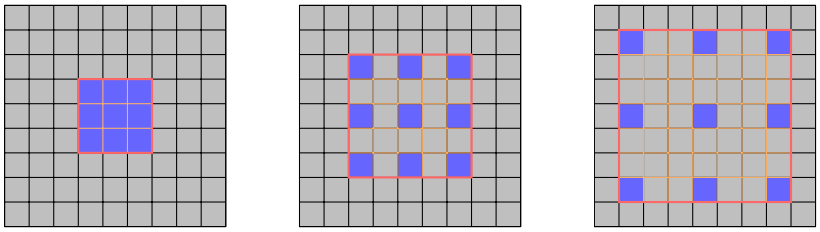


FIGURE 4.7 – Exemple de filtre de convolution "atrous" pour des filtres de taille 3×3. La convolution standard correspond à une convolution atrous avec un espacement de 1. Augmenter l'espacement élargit le champ réceptif de la convolution et permet un traitement multi-échelle [Chen et al., 2017b].

originale en une pré-segmentation grossière de faible résolution. Le réseau adapté par DeepLab est VGG-16 [Simonyan and Zisserman, 2015]. Pour construire la segmentation finale, les champs aléatoires conditionnels sont utilisés sur la sortie du réseau de neurones, au préalable redimensionnée à l'aide d'une interpolation bilinéaire.

En 2017, Liang-Chieh Chen propose une nouvelle version de DeepLab [Chen et al., 2017a], appelée DeepLab v2. Cette version reprend les mêmes principes que la précédente, en adaptant les réseaux VGG-16 [Simonyan and Zisserman, 2015] et ResNet-101 [He et al., 2016]. Cette fois cependant, le modèle se sert d'une pyramide spatiale de mise en commun à trous (en anglais *Atrous spatial pyramide pooling* ou ASPP) pour calculer la sortie du réseau de neurones, afin de traiter des objets à différentes échelles. Des champs aléatoires conditionnels sont toujours appliqués à cette sortie pour construire la segmentation finale.

Enfin, une troisième version de DeepLab est publiée (DeepLab V3) [Chen et al., 2017a]. Cette version utilise des convolutions atrous en cascade, en augmentant le taux de dilatation à chaque couche, puis en parallèle dans le module d'ASPP, pour saisir le contexte à différentes échelles. Le module d'ASPP est amélioré en incorporant des caractéristiques directement calculées à la manière de ParseNet [Liu et al., 2015a], à l'aide d'une couche de mise en commun globale. Ce bloc d'ASPP construit la segmentation finale et ce réseau n'utilise plus les champs aléatoires conditionnels.

Un autre réseau qui utilise les convolutions atrous est le réseau ENet [Paszke et al., 2016]. Comme SegNet [Badrinarayanan et al., 2017], il utilise une structure d'encodeur-décodeur. Les couches atrous sont situées dans la partie la plus profonde de l'encodeur. Ce réseau fonctionne en temps réel, c'est-à-dire, selon les auteurs, à plus de 10 images par seconde [Paszke et al., 2016].

ParseNet [Liu et al., 2015a] est une approche qui permet d'utiliser le contexte global pour réaliser la segmentation sémantique. Le contexte global est obtenu par une couche de mise en commun globale (en anglais *global-pooling*) sur les caractéristiques, ce qui permet de transformer les caractéristiques en un vecteur de contexte global. Ce vecteur

est ensuite dupliqué pour avoir les mêmes dimensions spatiales que les caractéristiques. Ces caractéristiques sont normalisés suivant la norme L2, puis le vecteur de contexte et les caractéristiques sont concaténés avant d'être utilisés par les couches suivantes du réseau.

BiSeNet est une architecture récemment introduite [Yu et al., 2018]. Les auteurs remarquent que la plupart des architectures sacrifient l'information spatiale, par *max-poolings* successifs, pour obtenir une vitesse de calcul raisonnable, ce qui grève les performances. Ils proposent donc un réseau de neurones "bilatéral". Un premier chemin permet de préserver l'information spatiale en générant des cartes de caractéristiques à très haute résolution, tandis qu'un autre chemin avec une approche de sous-échantillonnage rapide construit des caractéristiques basées sur le contexte. Un module de fusion permet de fusionner les sorties de chacun des chemins, pour agréger les information de localité et de contexte, afin de réaliser la segmentation finale. Dans une nouvelle version, le module de fusion est constitué d'une couche d'agrégation guidée, où les caractéristiques contextuelles influent sur la sélection des caractéristiques spatiales [Yu et al., 2020].

Les modèles basés sur une segmentation multi-résolution ou sur un traitement en parallèle de la localité et du contexte, bien qu'étant les plus performants de l'état de l'art, ne semblent pas adaptés à notre cas d'utilisation. En effet, même si ils peuvent être très rapides, ils contiennent un grand nombre de paramètres, ce qui peut les rendre difficile à entraîner dans notre cas, où nous ne disposons pas de vérités terrains précises. Ainsi, nous utilisons dans la suite des modèles plus simples, donc moins sujet au surapprentissage, inspirés d'une architecture encodeur-décodeur.

4.4 Première segmentation automatique

Un des défis du problème que nous essayons de résoudre est le besoin critique d'un grand nombre de données annotées. Il est bien connu que l'étiquetage des données est souvent une tâche difficile, requérant toujours beaucoup de temps humain et qui est donc coûteuse. C'est la raison pour laquelle nous avons décidé de nous appuyer sur une première segmentation automatisée mais bruitée pour entraîner notre modèle.

Comme nous le montrons dans la Section 4.5, l'entraînement de notre réseau est donc adapté à la nature brute de ce jeu de données, car nous ne pouvons pas nous attendre à ce qu'il fasse office d'une véritable vérité terrain.

Nous effectuons une première segmentation par un processus ad-hoc, adapté à notre configuration particulière. Le but de cette segmentation est de construire un jeu de données de segmentation bruité mais peu coûteux et de grande taille pour alimenter notre réseau de neurones afin de réaliser une segmentation par apprentissage. Par conséquent, cette segmentation n'a pas besoin d'être parfaite, mais elle doit être suffisamment précise pour

Algorithme 1 : Algorithme pour la segmentation automatique des données.

entrées : l'image de profondeur *image*,
le fond statique $fond$,
la zone sélectionnée *zone*,
le point de suivi *point*,
la surface minimale de l'opérateur *aire*
sortie : l'image segmentée
def *segmente(image, fond, zone, point, aire)* **:**
`/* Section 4.4.2 */`
Effacer $fond$ dans *image*
Supprimer le bruit dans *image* par ouverture

`/* Section 4.4.3 */`
Rassembler les objets proches dans *image* par fermeture
Détecter le contour des objets dans le masque
Effacer les objets dont la surface est inférieure à *aire*
si *il y a au moins un objet dans zone* **alors**
  Prendre l'objet dans *zone* dont le centre est le plus proche de *point*
sinon
  Prendre l'objet dont le centre est le plus proche de *point*
fin
Fixer *point* comme le centre de l'objet selectionné
Fixer *aire* comme la moitié de la surface de l'objet sélectionné
Segmenter *image* en utilisant l'objet comme un masque binaire
retourner *image segmentée*

être fiable en vue du traitement suivant.

Cette méthode nécessite une unique interaction avec l'utilisateur : on lui demande de sélectionner une zone quadrilatérale qui détermine la **zone d'évolution de l'opérateur**, c'est-à-dire une zone dans laquelle l'opérateur se trouve généralement lorsqu'il travaille, bien qu'il puisse occasionnellement en sortir. Le centre de cette zone est utilisé comme initialisation d'un point de suivi, qui est utilisé pour effectuer le suivi de la position de l'opérateur.

Il est également nécessaire de construire un **fond statique**, afin de supprimer de l'image de profondeur tous les objets fixes de la scène, qu'ils appartiennent à l'arrière plan (comme par exemple les autres tables de tri ou le mobilier présent dans la cabine) ou au premier plan (tels que la table de tri sur laquelle opère le valoriste ou les exutoires associés à son poste). Nous y parvenons en prenant l'image moyenne d'un ensemble d'images de profondeur de la scène vide, lorsque la chaîne de tri est à l'arrêt et que les opérateurs ne sont pas en cabine.

Le processus de segmentation automatique qui suit ces pré-traitements est découpé en deux étapes :

1. Nous supprimons les objets statiques à l'aide du fond statique ;

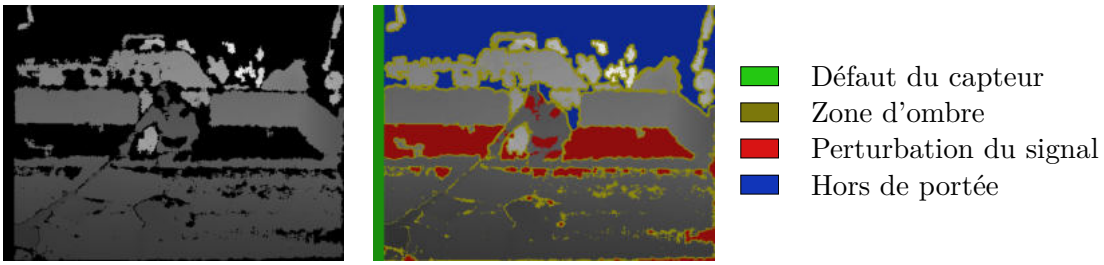


Image originale — Cause de l'absence d'information

FIGURE 4.8 – Image de profondeur brute (à gauche) et les différentes causes qui peuvent provoquer une absence d'information de profondeur pour un pixel de l'image de profondeur (à droite). On retrouve principalement : un défaut du capteur (en vert), un pixel hors de portée (en bleu), la présence d'une zone d'ombre (en jaune) ou d'un revêtement réfléchissant, d'une texture ou d'une couleur perturbant le signal (en rouge).

2. Nous sélectionnons l'opérateur humain parmi les objets restants.

Le pseudo-code de notre algorithme est donné dans l'Algorithme 1, et ces deux étapes sont décrites plus en détail dans la suite de cette partie.

4.4.1 Pixels vides dans l'image de profondeur

Avant de décrire en détail le procédé de segmentation des images de profondeur, commençons par étudier les pixels vides dans l'image de profondeur. Les pixels vides de l'image sont des pixels pour lesquels le capteur n'a pas été en mesure de déterminer la profondeur. Dans notre cas, de tels pixels apparaissent généralement dans quatres cas (illustrés en Figure 4.8) :

— un **défaut du capteur** : les pixels concernés sont toujours vides ;
— la partie de la scène représentée par ce pixel est **hors de portée** du capteur, elle peut être trop éloignée ou trop proche ;
— la partie de la scène représentée par ce pixel se trouve dans une **zone d'ombre**, c'est-à-dire qu'elle n'est pas visible par l'émetteur et le récepteur du capteur ;
— le signal qui couvre la partie de la scène représentée par ce pixel est perturbé par la présence d'un **revêtement réfléchissant**, une **matière brillante** ou un **objet de couleur noire**.

Certains de ces défauts dans l'image ne sont pas gênants dans notre cas d'utilisation, car ils se produisent loin de la zone d'activité de l'opérateur, comme les pixels vides à cause d'un défaut du capteur, qui apparaissent au niveau des limites de l'image, ou ceux qui sont hors de portée du capteur, mais d'autres, comme la présence d'éléments perturbateurs ou de zone d'ombre, peuvent grandement modifier le traitement de l'image, en particulier lorsque des bandes réfléchissantes sur les vêtements des opérateurs les découpent en plusieurs

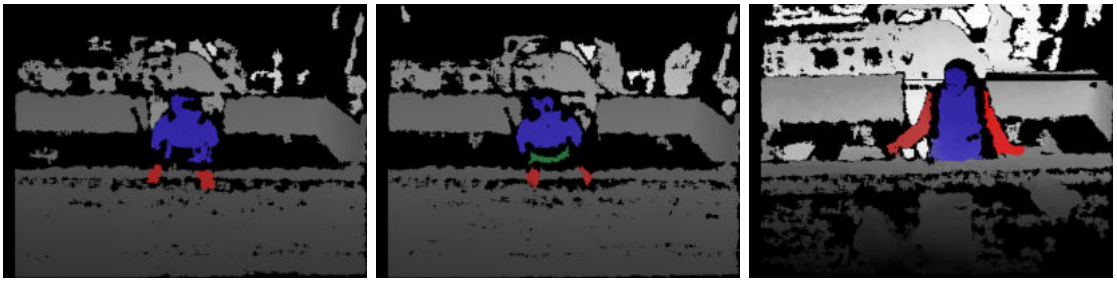


FIGURE 4.9 – Exemples de découpe de l'opérateur causée par la présence de bandes réfléchissantes qui perturbent le signal, avec deux capteurs et deux opérateurs différents. On observe qu'en fonction des bandes réfléchissantes présentes sur les vêtements, les parties du corps ne sont pas séparées de la même façon (gauche et droite). De plus, on peut avoir des découpes différentes en fonction de la posture du valoriste (gauche et milieu).

parties (voir Figure 4.9).

4.4.2 Suppression des objets statiques

La première étape consiste à supprimer les objets statiques de la scène de l'image de profondeur.

Nous débutons notre suppression du fond sur les images de profondeur en comparant, pour chaque pixel, sa valeur de profondeur sur l'image du fond statique et sa valeur sur l'image à segmenter. En raison de l'incertitude au niveau de la mesure de profondeur du capteur, on considère qu'un pixel fait partie du fond si la différence de profondeur entre le fond statique et l'image de profondeur à segmenter n'excède pas 2%. Ce pourcentage peut sembler important à première vue, car les constructeurs des différents capteurs de profondeur ne rapportent qu'une erreur de l'ordre d'un à deux centimètres pour un objet situé à deux mètres de distance (soit 1% d'écart). Cependant, dans notre cas d'usage, cet écart est plus important et quelques légères vibrations peuvent parfois faire varier la profondeur d'un pixel. Fixer un seuil à 3% permet de s'assurer une marge de sécurité, tout en minimisant le nombre de pixels de l'opérateur identifiés comme des pixels statiques. Toutefois, un inconvénient de ce seuil est que, lorsque les mains sont posées sur la table de tri, elles sont identifiées comme faisant partie du fond.

Une dernière difficulté consiste à effacer les pixels du fond statique, mais dont la valeur de profondeur sur l'image est éloignée de la valeur de profondeur moyenne du pixel. Ces pixels sont très souvent localisés au niveau des frontières entre les différents objets fixes, c'est-à-dire dans des zones d'ombre, ou sur un revêtement qui perturbe le calcul de la profondeur. La valeur de profondeur de ces pixels est souvent très volatile :

— si ce sont des pixels dans des zones d'ombres, ils peuvent prendre comme valeur la profondeur de l'objet qui fait de l'ombre, ou celle de l'objet caché ;

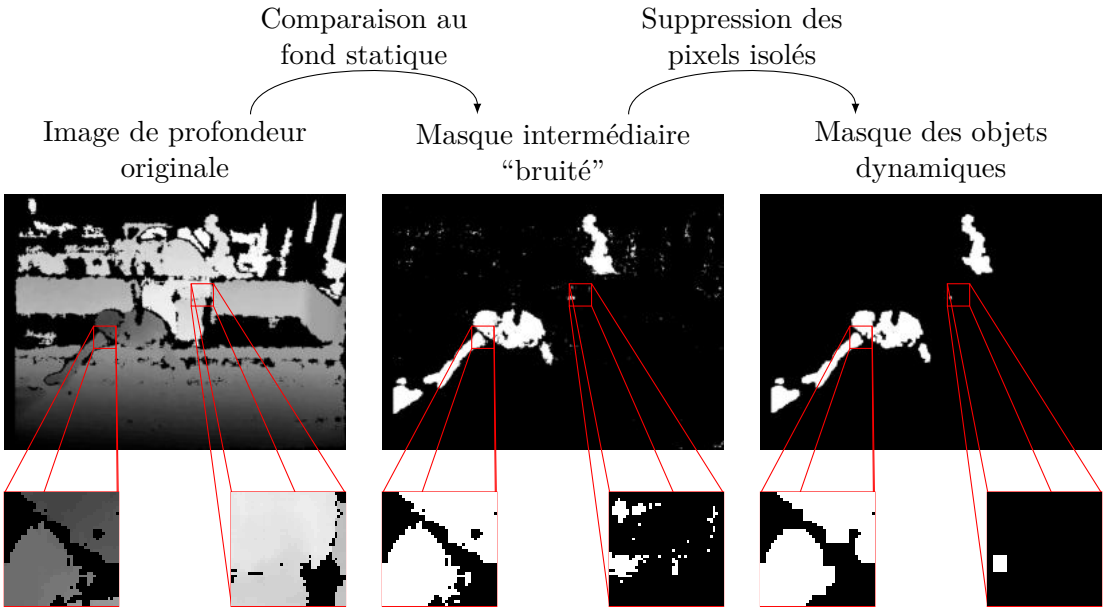


FIGURE 4.10 – Étape de suppression du fond statique de la segmentation automatique. Tout d'abord, on supprime la très grande majorité du fond statique en comparant directement les valeurs de profondeur à une image du fond statique préalablement acquise, puis on supprime les pixels isolés restant à l'aide d'une transformation morphologique d'ouverture. On observe que cette suppression du fond conserve cependant l'écart entre les différentes parties du corps du valoriste, lorsqu'elles sont séparées.

— si ce sont des pixels sur des surfaces qui perturbent le signal, la valeur de profondeur n'est pas stable.

Pour effacer ces pixels, souvent relativement isolés, on applique une transformation morphologique d'ouverture, qui consiste en une érosion, puis une dilatation. L'érosion a pour but d'effacer les pixels isolés, alors que la dilatation reconstruit la surface des objets plus grands dont la frontière a été supprimée par l'opération d'érosion. Cette transformation est appliquée avec un noyau de taille 5×5. L'intérêt de cette suppression est qu'elle permet d'effacer des pixels proches de la silhouette de l'opérateur, mais qui n'en font pas partie. Ce sont des pixels qui auraient pu être considérés comme appartenant à l'opérateur lors de l'étape suivante, et qui peuvent considérablement fausser les résultats.

Grâce à ces opérations, illustrées en Figure 4.10, les objets statiques de la scène ont été effacés de l'image de profondeur, et on obtient donc un masque des objets dynamiques. Il reste maintenant à trouver, parmi les objets en mouvement restant, lequel correspond à l'opérateur analysé.

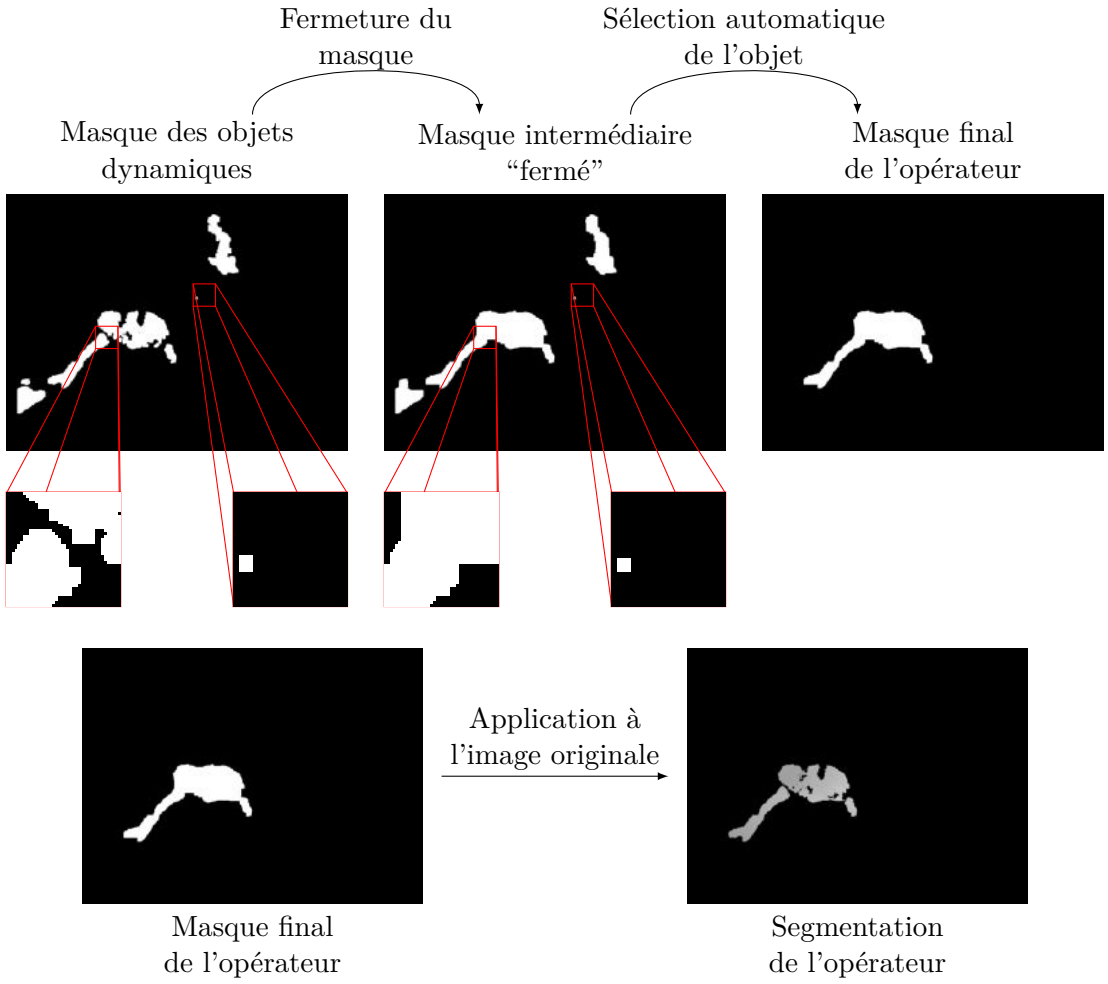


FIGURE 4.11 – Étape de sélection de l'opérateur lors de la segmentation automatique. On ferme le masque des objets dynamiques précédemment obtenu pour rassembler les différentes parties de l'opérateur, puis on sélectionne l'objet qui représente le valoriste. Ceci nous permet d'obtenir un masque final, qui, appliqué sur l'image de profondeur originale, nous donne la segmentation de l'opérateur.

4.4.3 Sélection de l'opérateur parmi les objets en mouvement

La deuxième étape de notre algorithme de segmentation consiste à retrouver l'opérateur parmi les objets en mouvement, qui sont encore tous présents sur l'image de profondeur. Elle est illustrée en Figure 4.11.

Avant d'effectuer cette sélection, on recourt à une nouvelle transformation morphologique afin de rassembler les parties potentiellement séparées du corps du valoriste. Cette fois-ci, on utilise une fermeture, qui consiste en une dilatation puis une érosion, et permet de raccorder des éléments proches. On utilise maintenant un noyau de dimension sensiblement plus importante (15×15), car la séparation entre les membres peut faire jusqu'à

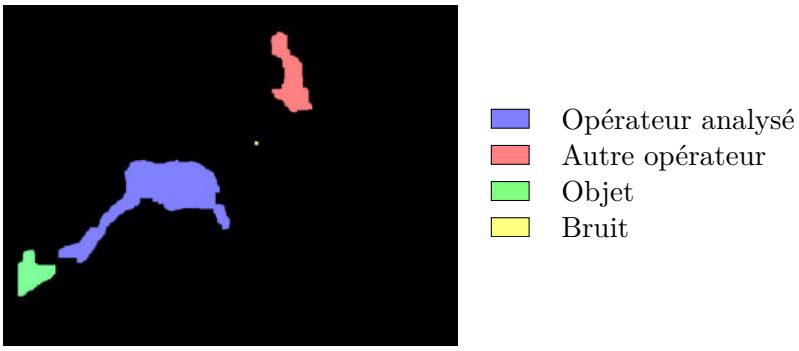


FIGURE 4.12 – Les différents objets qui restent sur le masque des objets dynamiques, une fois la suppression du fond statique et la fermeture du masque réalisée. En plus de l'opérateur monitoré (en bleu), on peut retrouver quelques petits éléments de bruit (en jaune), d'autres opérateurs en cabine (en rouge) ainsi que les objets qui circulent sur la table (en vert). Une analyse de ce masque et une sélection de la silhouette correspondant à l'opérateur est donc nécessaire.

trente pixels.

Une fois les différentes parties regroupées, on cherche à déterminer quel objet du masque représente l'opérateur. En effet, comme le montre la Figure 4.12, il reste encore plusieurs éléments distincts sur le masque de segmentation. L'objectif est de trouver l'objet qui correspond à l'opérateur en utilisant l'aire et la position de chacun des objets.

Afin d'établir une liste des objets, nous détectons donc les contours présents sur le masque à l'aide d'un détecteur de contour basé sur l'algorithme proposé par [Suzuki and Abe, 1985]. Cet algorithme, utilisé sur des images binaires, balaie l'image pour détecter les frontières des objets, puis circule le long de ces frontières pour construire le contour complet des objets. Il assigne à chaque frontière distincte une valeur unique, ce qui permet de distinguer les objets et de les compter. Suite à cet algorithme, nous obtenons une liste de contours qui définissent chacun un objet candidat.

Une fois cette liste de contours construite, nous nous en servons pour sélectionner la forme qui correspond à l'opérateur. Pour ce faire, nous comparons tout d'abord l'aire de chacun des objets détectés à une aire minimale qui représente 50% de l'aire de l'opérateur identifié sur la dernière image pour laquelle le valoriste était situé à l'intérieur de la zone. Ceci nous permet d'exclure les objets considérés comme trop petits. Cependant, il n'y a pas de limite haute à l'aire d'un opérateur : en effet, lorsque l'opérateur tient de la matière, il est très difficile de le séparer de cette matière, dont l'aire vient s'additionner à celle du valoriste. Ensuite, on recherche les objets qui se trouvent à l'intérieur de la zone sélectionnée par l'utilisateur, en utilisant leur barycentre : on choisit l'objet dont le barycentre est le plus proche du point de suivi. Si aucun objet n'est situé à l'intérieur de la zone, on prend l'objet situé à l'extérieur dont le barycentre est aussi le plus proche du

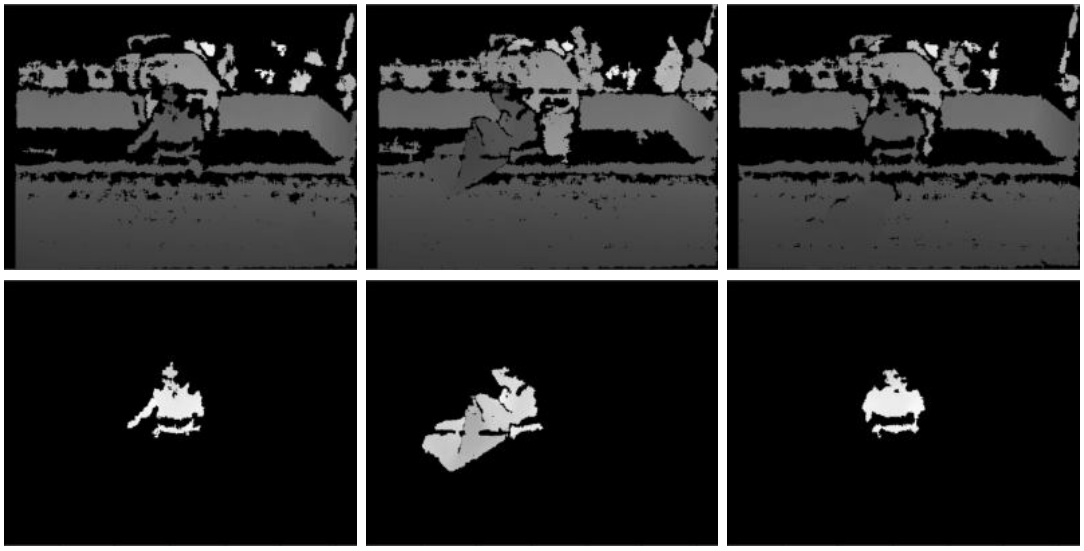


FIGURE 4.13 – Exemples d'images de notre jeu de données (ligne du haut) et leur segmentation automatique associée obtenue grâce à notre algorithme (en bas). Si certaines segmentations sont très bonnes (à gauche), la présence de gros objets peut polluer les résultats (au milieu). De plus, les mains peuvent manquer (à droite).

point de suivi.

Lorsque nous segmentons les images consécutives d'une vidéo de profondeur, nous mettons à jour à la fois la position du point de suivi et la taille minimale de l'objet pour la segmentation de l'image suivante.

Cet algorithme nous permet d'extraire une première segmentation de l'opérateur à partir des images de profondeur. Comme on peut facilement le comprendre, cette segmentation n'est pas robuste : un exemple d'erreur fréquent est qu'un objet qui est détenu par l'opérateur est presque toujours extrait comme faisant partie de l'opérateur, comme le montre la Figure 4.13. C'est cependant souvent le cas lorsqu'on cherche à segmenter des images de profondeur, comme expliqué en Section 3.2.6.

Suite à cette segmentation automatique, notre jeu de données contient 59 384 images et leur segmentation associée. Des exemples de cet ensemble de données sont présentés en Figure 4.13.

4.5 Entraînement itératif avec filtrage des données

La segmentation proposée par l'algorithme de segmentation automatique est de qualité très hétérogène : certains masques segmentent parfaitement l'opérateur, alors que dans

d'autres cas, la segmentation automatique est totalement fausse. Des segmentations de qualité intermédiaire sont aussi obtenues, où l'opérateur est segmenté correctement, mais l'objet est aussi inclus dans le masque. Ainsi, afin de corriger ces erreurs, parfois grossières, nous décidons de recourir à une méthode d'apprentissage qui s'appuie sur cette segmentation hétérogène pour construire une segmentation de meilleure qualité. Nous choisissons d'utiliser un réseau de neurones, dont les données d'apprentissage, qui sont en fait les segmentations obtenues précédemment, seront itérativement filtrées pour ne conserver que celles de bonne qualité. Nous espérons que la capacité de généralisation des réseaux de neurones nous permettra d'améliorer la qualité globale des segmentations.

Le contexte de cette thèse, où les systèmes développés sont utilisés dans un environnement industriel, demande à ce que nous recourrions à des architectures légères pour obtenir des performances en temps réel. Il est par exemple souhaité que nous minimisions le nombre d'ordinateurs et la puissance de ceux-ci pour effectuer les analyses, afin d'optimiser le coût de mise en place et d'utilisation du dispositif. Ainsi, nous choisissons de redimensionner les images de 640×480 en images 64×64. Ce changement permet par ailleurs de résoudre un problème sous-jacent de gestion de l'espace mémoire pour charger l'ensemble de données. En effet, nos données, où la profondeur est échantillonnée sur 16 bits, sont plus volumineuses que des images en niveau de gris échantillonnées sur 8 bits (256 valeurs), car on souhaite conserver la valeur précise de la profondeur de chaque pixel, donnée en millimètres.

4.5.1 Architecture des réseaux de neurones

Nos expérimentations ont été menées en utilisant deux modèles de réseaux de neurones inspirés de l'état de l'art : SegNet [Badrinarayanan et al., 2017] et U-net [Ronneberger et al., 2015]. Ces deux réseaux, ayant pour objectif de réaliser la segmentation sémantique d'images, ont auparavant fait leurs preuves lorsqu'ils ont été utilisés sur des images couleur. Les résultats obtenus semblent montrer que l'approche est généralisable à tout type de réseau de neurones.

Cependant, il est important de noter que notre contribution ne réside pas dans l'architecture de ces réseaux, mais surtout dans la façon dont nous adaptons l'entraînement de ces réseaux en raison de la qualité variable de l'ensemble des données d'apprentissage.

Réseau inspiré de SegNet

Notre premier réseau de neurones est un encodeur-décodeur directement inspiré de SegNet introduit par [Badrinarayanan et al., 2017]. Comme nous traitons des images à la fois en entrée et en sortie, c'est un réseau de neurones entièrement convolutif. L'architecture du réseau est illustrée en Figure 4.14. Le modèle est décomposé en deux parties :

— la partie encodeur : elle contient deux paires de couches convolutives séparées par

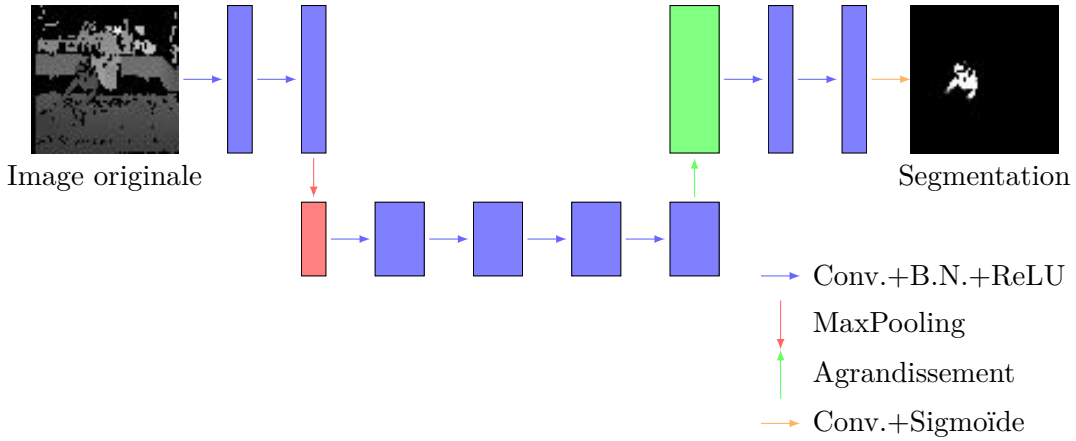


FIGURE 4.14 – Architecture du premier réseau de neurones étudié, directement inspiré de SegNet [Badrinarayanan et al., 2017]. Le réseau de neurones est un modèle entièrement convolutif de type encodeur-décodeur. Chaque couche de convolution est constituée de filtres de taille 3×3, puis suivie d'une couche de *Batch Normalization* (B.N.) et d'une fonction d'activation ReLU. La première couche convolutive contient 64 filtres et ce nombre double après le *max-pooling* et est divisé par deux après l'agrandissement. Les *max-pooling* et les agrandissements utilisent des noyaux de taille 2×2.

une couche de *max-pooling*. Nous appliquons une couche de *batch normalization* et une activation de type ReLU après chaque couche convolutive. La première couche convolutive contient 64 filtres de taille 3×3, et le nombre de filtres double après la couche de *max-pooling* ;

— la partie décodeur : elle suit le schéma miroir de la partie encodeur, avec les couches de *max-pooling* remplacées par des couches d'agrandissement (en anglais *upsampling*). Le nombre de filtres des couches de convolution est divisé par deux après la couche d'agrandissement.

Une dernière couche convolutive contenant un unique filtre et suivie d'une activation sigmoïde permet de construire un masque contenant la probabilité de chaque pixel d'appartenir ou pas à l'opérateur.

Réseau inspiré de U-net

Notre second réseau de neurones est un encodeur-décodeur directement inspiré de U-Net introduit par [Ronneberger et al., 2015]. C'est une nouvelle fois un réseau de neurones entièrement convolutif. L'architecture du réseau est illustrée en Figure 4.15. L'architecture du modèle reprend celle du réseau de neurones présenté précédemment et est de nouveau décomposée en deux parties :

— la partie encodeur : elle contient deux paires de couches convolutives séparées par une couche de *max-pooling*. Nous appliquons une couche de *batch normalization* et

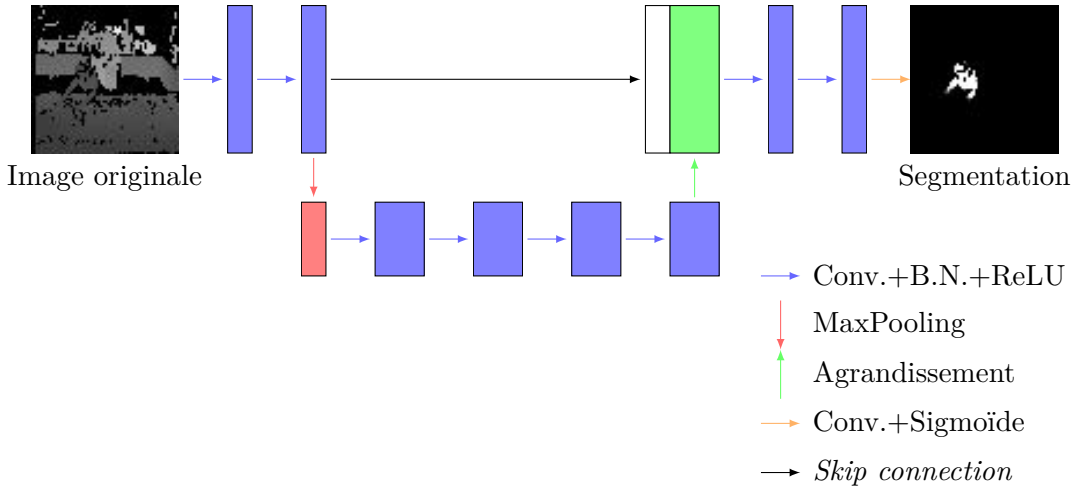


FIGURE 4.15 – Architecture du second réseau de neurones étudié, directement inspiré de U-net [Ronneberger et al., 2015]. Comme SegNet [Badrinarayanan et al., 2017], ce réseau de neurones est un modèle entièrement convolutif de type encodeur-décodeur, dont l'architecture est fortement inspirée du réseau précédent (voir Figure 4.14). La différence avec le réseau inspiré de SegNet se fait au niveau de l'ajout d'une *skip connection* qui concatène la sortie de la dernière couche de convolution du premier niveau au résultat de l'opération d'agrandissement du décodeur.

une activation de type ReLU après chaque couche convolutive. La première couche convolutive contient 64 filtres de taille 3×3, et le nombre de filtres double après la couche de *max-pooling* ;

— la partie décodeur : elle suit le schéma miroir de la partie encodeur, avec les couches de *max-pooling* remplacées par des couches d'agrandissement (en anglais *upsampling*). Le nombre de filtres des couches de convolution est divisé par deux après la couche d'agrandissement.

La différence avec le modèle SegNet [Badrinarayanan et al., 2017] vient du fait que l'on a ajouté une opération de *skip connection* qui concatène la sortie de la dernière couche convolutive de la première paire de l'encodeur à la sortie de l'opération d'agrandissement du décodeur.

Comme précédemment, une dernière couche convolutive contenant un unique filtre et suivie d'une activation sigmoïde permet de construire un masque contenant la probabilité de chaque pixel d'appartenir ou pas à l'opérateur.

Hyper-paramètres pour la phase d'entraînement

Pour entraîner nos réseaux, nous utilisons l'optimiseur Adam avec ses paramètres par défaut et l'entropie binaire croisée comme fonction de coût. En effet, l'optimiseur Adam a l'avantage d'être un optimiseur à taux d'apprentissage adaptatif et donc de converger plus

rapidement et efficacement qu'une simple descente de gradient stochastique. L'entropie croisée est la fonction de coût généralement utilisée pour les tâches de classification et de segmentation. Dans dans notre cas, nous utilisons la version binaire de cette fonction, car nous ne cherchons qu'à distinguer deux classes.

4.5.2 Mise à jour de l'ensemble d'entraînement

Comme notre ensemble de données d'apprentissage ne peut pas être considéré comme une vérité terrain, nous introduisons une méthode pour sélectionner automatiquement des données de qualité acceptable, et nous mettons à jour notre ensemble de données en conséquence.

Contrairement à l'apprentissage actif [Tuia et al., 2011], où il est demandé à un expert de (ré)étiqueter les données en fonction des prédictions du réseau de neurones, nous proposons une méthode automatique réutilisant uniquement le jeu de données fourni, et sans intervention d'un expert.

On observe que la qualité de la segmentation obtenue par l'algorithme automatique présenté dans la Section 4.4 est très variable selon les images. Certaines segmentations sont presque parfaites, tandis que d'autres sont totalement fausses. Le défi consiste donc à séparer correctement le bon grain de l'ivraie de manière automatisée. Nous cherchons à nous servir de la capacité de généralisation des réseaux de neurones pour calculer la qualité d'une segmentation automatique : dans le cas où l'on possède une majorité de données correctement annotées, si le réseau de neurones réalise des prédictions différentes de la donnée annotée qu'on utilise pour lui faire apprendre à segmenter, c'est que cette annotation comporte probablement une aberration. Nous décidons donc de nous fier uniquement aux résultats de l'entraînement du réseau de neurones : en comparant la sortie du modèle à celle attendue, c'est-à-dire la segmentation automatique construite par l'algorithme présenté en Section 4.4, on cherche à séparer les annotations de bonnes qualités du reste de l'ensemble de données.

Nous choisissons d'utiliser le coefficient de similarité Jaccard pour évaluer la similarité entre la segmentation prédite par le réseau et la segmentation donnée dans l'ensemble de données pour toutes les images contenues dans le jeu de données d'entraînement, à chaque étape de mise à jour, que cette donnée ait été utilisée pour l'entraînement ou non.

Le coefficient de similarité Jaccard J entre deux régions A et B, aussi appelé "intersection sur union" (en anglais *intersection over union*, abrégé IoU), est simplement la taille de l'intersection divisée par la taille de l'union des deux masques de segmentation :

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \in [0,1] \tag{4.1}$$

Ainsi, le coefficient de Jaccard est proche de 0 lorsque les deux masques de segmentation sont très différents, et proche de 1 lorsque qu'ils se superposent. On cherche donc à

Approche	Minimum	Médiane	Moyenne	Maximum
Segmentation automatique	0.000	0.814	0.762	**1.000**
SegNet	0.033	0.728	0.691	**1.000**
U-net	0.000	0.682	0.668	0.990
SegNet avec mise à jour	0.291	**0.825**	**0.807**	**1.000**
U-net avec mise à jour	**0.363**	0.822	0.801	**1.000**

TABLE 4.2 – Coefficient de Jaccard de nos différentes segmentations par rapport à la vérité terrain après 7 époques. Comme attendu, l'utilisation du jeu de données complet pour entraîner le réseau n'améliore pas les résultats. En revanche, l'apprentissage avec l'approche proposée et la mise à jour du jeu de données améliorent la qualité des segmentations.

maximiser la valeur de ce coefficient.

Nous décidons de mettre à jour l'ensemble de données et de supprimer les données les plus mauvaises pendant la phase d'apprentissage du réseau. Notre idée vient du fait que l'on observe que le réseau de neurones est très rapidement capable de fournir une intuition de la segmentation correcte de l'image (illustrée en Figure 4.16), et que l'apprentissage ultérieur, qui pourrait conduire à un surajustement, affine cette intuition vers la segmentation attendue, dans notre cas en ajoutant ce que nous voyons parfois comme du bruit, tels que les objets tenus par l'opérateur, pour qu'il corresponde aux annotations de l'ensemble d'entraînement.

Nous utilisons cette intuition de la segmentation par le réseau de neurones pour supprimer les pires données de l'ensemble de données d'apprentissage pendant les premières étapes de l'entraînement. Comme le montre la Section 4.6, cela permet d'améliorer les résultats du réseau. Nous ne remplaçons pas les données supprimées par les sorties du réseau de neurones, car nous pensons que ces sorties sont généralement encore trop grossières pour être utilisées comme annotations pour l'apprentissage, comme montré en Figure 4.16.

4.6 Expérimentations et évaluation

Afin de quantifier l'impact de notre politique d'apprentissage, nous avons annoté manuellement 520 images à l'aide du logiciel fourni par [McGuinness and O'connor, 2010], et nous considérons ces images comme des vérités terrains de notre segmentation. Nous avons soigneusement choisi cet ensemble d'images pour qu'il affiche un large éventail de postures prises par l'opérateur, dans des conditions variées et difficiles : passage d'un autre opérateur dans l'arrière plan, préhension d'objets, etc. Nous utilisons ces données pour mesurer la performance de nos algorithmes (voir Figure 4.3 pour des exemples). La comparaison entre la vérité terrain et les segmentations est donnée dans le Tableau 4.2 et la Figure 4.18.

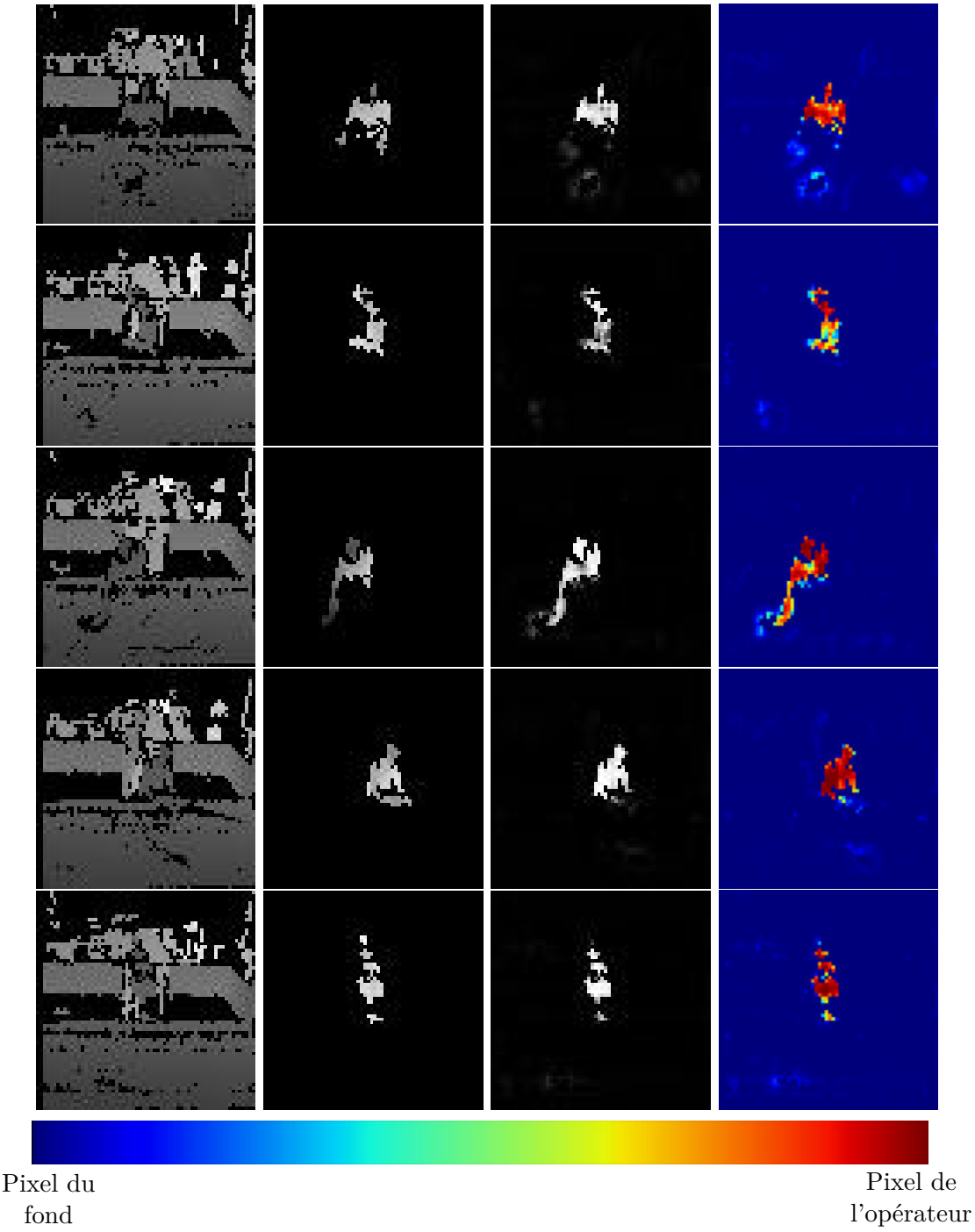


FIGURE 4.16 – Prédictions du modèle pour la segmentation après un entraînement d'une époque. Colonne de gauche : entrée du réseau de neurones. Deuxième colonne : vérité terrain. Troisième colonne : prédiction du réseau de neurones. Dernière colonne : carte de probabilités d'appartenance d'un pixel à l'opérateur en sortie du réseau de neurones. On observe que le réseau de neurones fournit déjà une bonne segmentation (Jaccard de 0.738), mais que certains morceaux de bras sont manquants, ou certains objets tenus déjà mal classés. Cependant, on peut voir sur les cartes de probabilités que le réseau de neurones est en général très confiant pour les pixels appartenant à l'opérateur ou au fond statique.

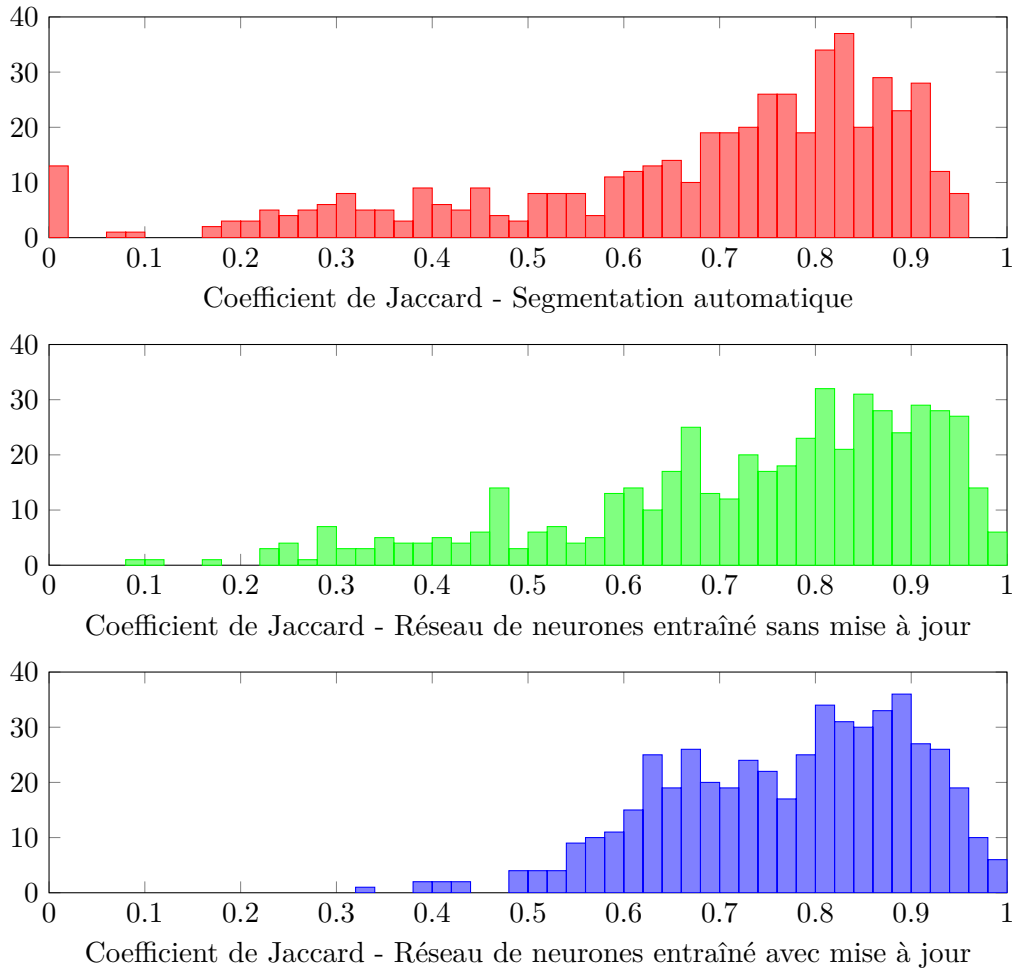


FIGURE 4.17 – Histogrammes du coefficient de Jaccard après segmentation automatique (en haut, en rouge), entraînement du réseau inspiré de SegNet sans mise à jour (au milieu, en vert) et avec mise à jour (en bas, en bleu). On observe que la répartition des données est meilleure pour l'apprentissage avec mise à jour de l'ensemble d'entraînement : dans ce cas, la majorité des données a un Jaccard compris entre 0.75 et 1 (on a exactement 307 images avec un coefficient supérieur à 0.75).

Les résultats de la segmentation sont présentés en Figure 4.19 et en Figure 4.20.

Comme on peut le voir, notre segmentation bruitée se comporte généralement bien, avec une valeur moyenne de 0.762 pour son coefficient de Jaccard. Le principal problème est que même si elle est parfaite dans certains cas (avec un Jaccard de 1.0), elle peut complètement échouer dans d'autres cas (coefficient de Jaccard de 0.0). En outre, et comme prévu, l'entraînement de réseaux de neurones issus de l'état de l'art tels que SegNet et U-net à l'aide de cet ensemble de données n'améliore pas les résultats : la valeur médiane et la valeur moyenne de Jaccard diminuent toutes les deux (valant alors respectivement 0.673 et 0.691 pour SegNet et 0.682 et 0.668 pour U-net). On en déduit donc que l'entraînement des

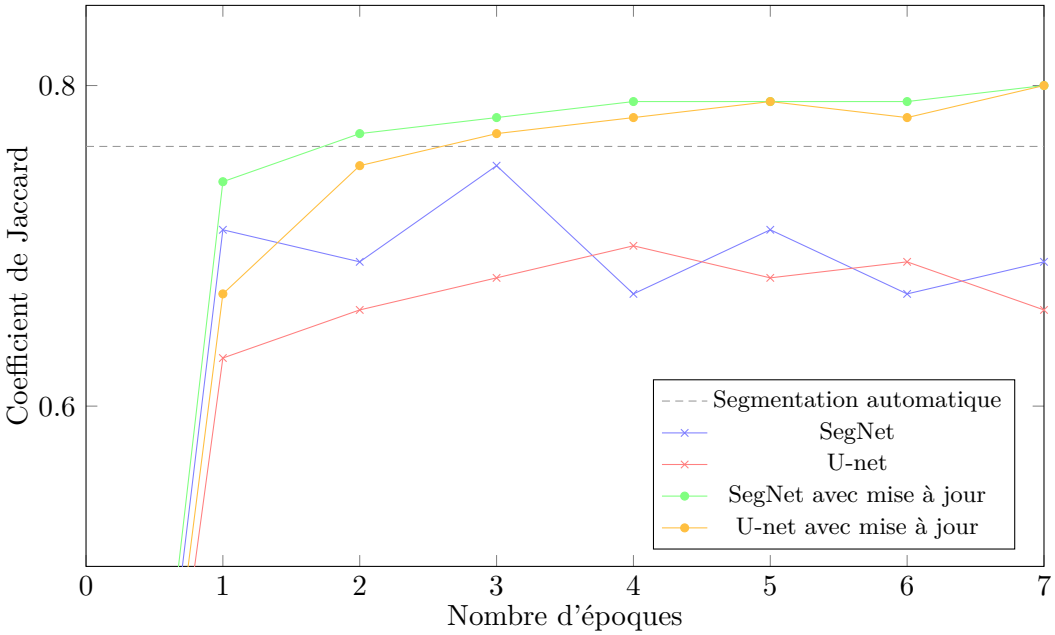


FIGURE 4.18 – Évolution du Jaccard pendant l'entraînement. L'apprentissage avec l'ensemble de données d'apprentissage complet reste sous les performances de la segmentation automatique, alors que l'apprentissage avec mise à jour de l'ensemble de données les surpasse.

réseaux de neurones est affecté par la trop grande proportion de mauvaises segmentations dans notre ensemble de données.

Le Tableau 4.2 et les histogrammes présentés en Figure 4.17 montrent que la mise à jour de l'ensemble de d'apprentissage améliore non seulement les résultats par rapport à l'apprentissage habituel, mais aussi les résultats donnés par la segmentation bruitée, ce qui est notre objectif. Ainsi, la valeur moyenne du coefficient de Jaccard augmente pour les deux modèles, passant de 0.691 à 0.807 pour le modèle SegNet et de 0.668 à 0.801 pour U-net, soit une amélioration respective de 16.7% et 19.9%.

Pour obtenir ces résultats, nous avons choisi de mettre à jour notre ensemble d'entraînement à la fin de chaque époque, en ne conservant à chaque itération que la moitié de l'ensemble de données complet ayant le Jaccard le plus haut.

Notre approche de mise à jour de l'ensemble d'entraînement ne dépend que deux hyperparamètres : la quantité de données sélectionnées, et la fréquence de mise à jour.

Nous avons mené de nombreuses expériences pour régler correctement ces deux hyperparamètres de mise à jour. La principale information donnée par ces expériences est qu'ils dépendent fortement du modèle, en particulier de la rapidité de sa convergence, et de la qualité de l'ensemble de données bruitées. Il n'y a donc pas de paramètre miracle. Un point essentiel est toutefois de mettre à jour l'ensemble de données pendant les premières époques de l'apprentissage (comme visible en Figure 4.22), et de ne pas être trop laxiste

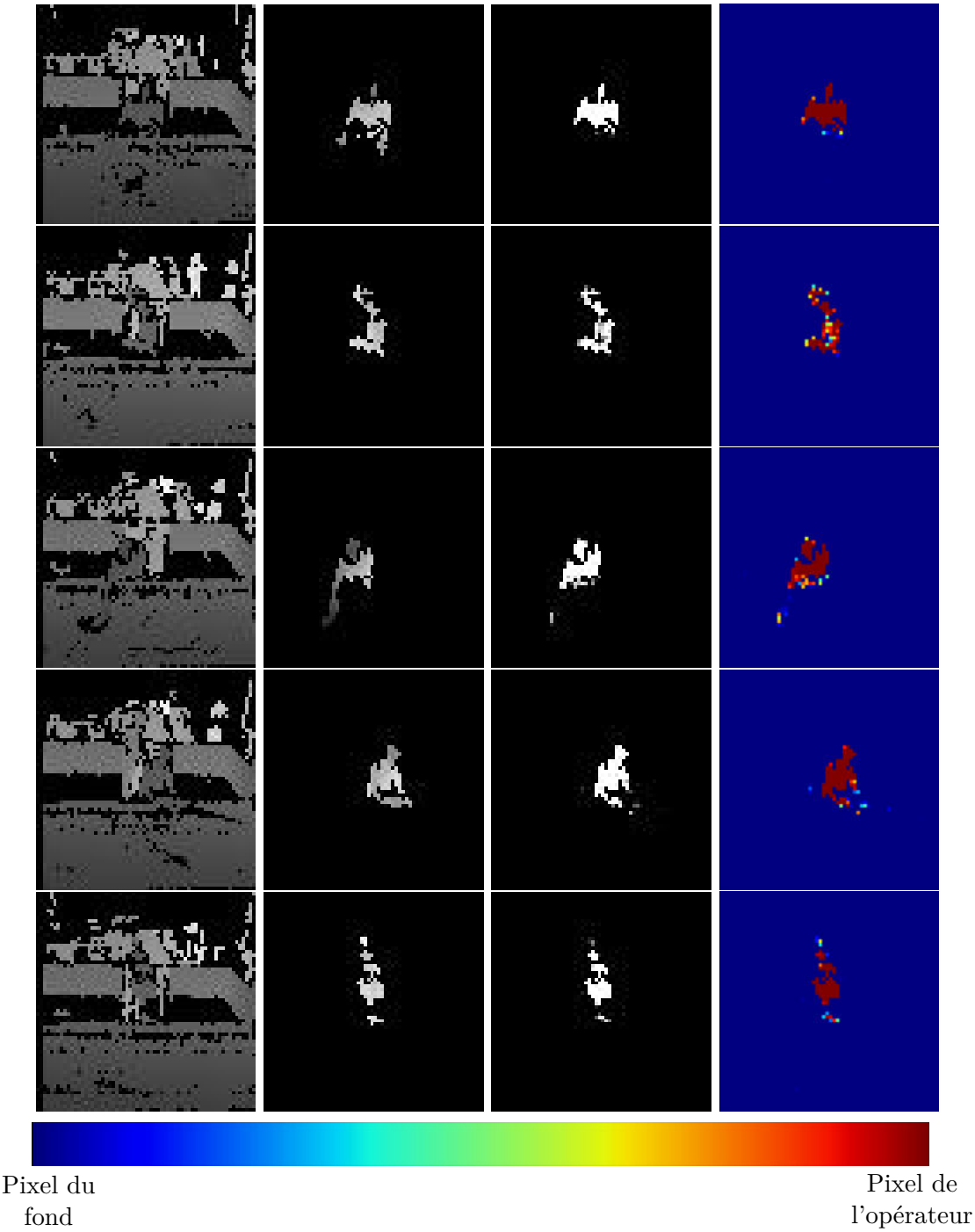


FIGURE 4.19 – Prédictions du modèle pour la segmentation après 7 époques. Colonne de gauche : entrée du réseau de neurones. Deuxième colonne : vérité terrain. Troisième colonne : prédiction du réseau de neurones. Dernière colonne : carte de probabilités d'appartenance d'un pixel à l'opérateur en sortie du réseau de neurones. On observe que les segmentations proposées par le réseau de neurones sont très proches des segmentations attendues, même si il manque parfois encore les mains de l'opérateur. On voit sur les cartes de probabilité que le réseau de neurones est en général très confiant dans la classification des pixels.

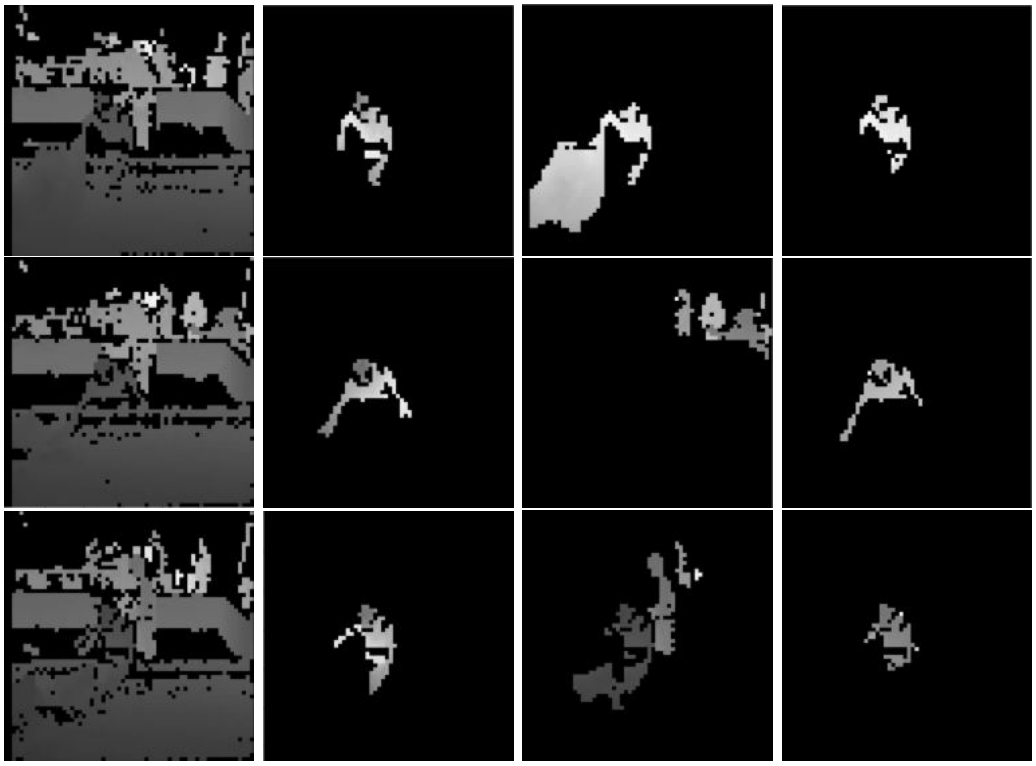


FIGURE 4.20 – Résultats obtenus par notre réseau comparés à la vérité terrain et à la segmentation automatique. Colonne de gauche : entrée du réseau de neurones. Deuxième colonne : vérité terrain. Troisième colonne : sortie de la segmentation automatique. Dernière colonne : sortie du réseau de neurones. Le réseau de neurones enlève des objets des mains de l'opérateur (première ligne) et donne une segmentation satisfaisante dans les cas où la segmentation automatisée échoue (deuxième et troisième lignes).

dans la sélection des données (voir Figure 4.21).

4.7 Conclusion

Nous proposons une méthode pour fournir une segmentation de bonne qualité de l'opérateur à partir d'images en profondeur en utilisant un réseau de neurones entraîné sur un jeu de données bruitées. Nous obtenons ce jeu de données en utilisant un algorithme de segmentation automatisé ad-hoc. Nous l'utilisons ensuite pour entraîner notre réseau de neurones. Pendant l'apprentissage, nous mettons à jour les données d'entraînement plusieurs fois pour éliminer les données les plus imparfaitement segmentées, en utilisant les premières sorties du réseau de neurones.

Notre schéma d'apprentissage original permet au réseau de mieux apprendre et nous obtenons un résultat moyen de 80,7% sur notre ensemble de données, alors que la procédure d'apprentissage habituelle n'atteint que 69,1%.

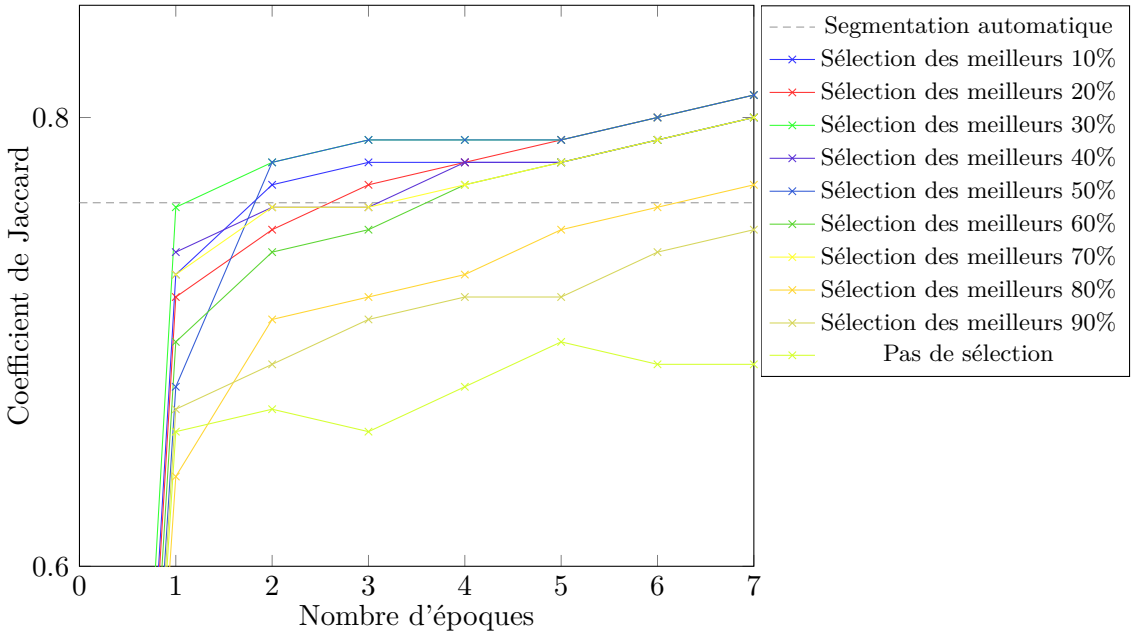


FIGURE 4.21 – Évolution du Jaccard pendant l'entraînement, en faisant varier la quantité de données sélectionnées. On observe que, dans la plupart des cas, sélectionner les meilleures données permet d'obtenir de meilleures performances que la segmentation automatique (de 10 à 80%), avec les meilleurs résultats lorsqu'on conserve 10 à 70% des données, où l'on obtient un Jaccard supérieur à 0.8.

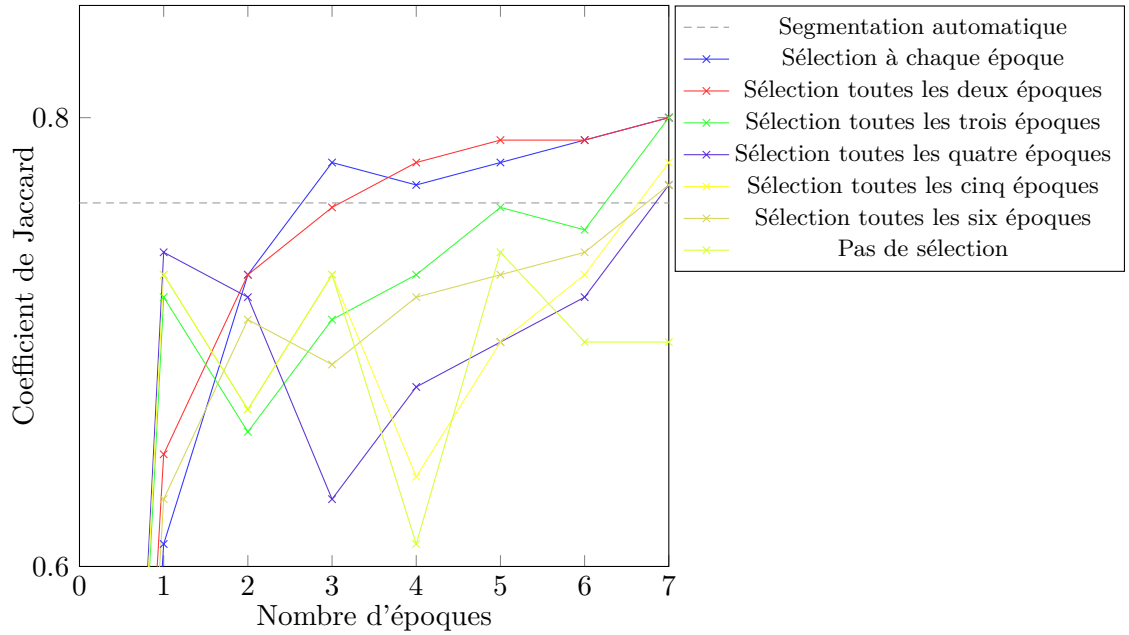


FIGURE 4.22 – Évolution du Jaccard pendant l'entraînement, en faisant varier la fréquence de mise à jour de l'ensemble d'entraînement. On observe que sélectionner les données plus tôt permet d'obtenir de meilleures performances.

En résumé

La première contribution présentée dans cette thèse est l'introduction d'un procédé d'apprentissage faiblement supervisé, avec mise-à-jour des données d'apprentissage automatique, dans l'objectif de réaliser la segmentation d'images de profondeur, afin d'extraire l'opérateur humain de celles-ci.

Dans un premier temps, nous présentons quelques concepts et travaux sur lesquels se sont appuyés les travaux présentés dans ce chapitre. Premièrement, nous décrivons dans cette section les méthodes d'apprentissage qui peuvent être utilisées lorsqu'on possède peu de données, ou des données imparfaites (voir Section 4.2). Deuxièmement, nous exposons en Section 4.3 les approches de segmentation d'image basées sur l'apprentissage profond. Nous nous sommes inspirés de certaines de ces approches pour construire nos modèles de segmentation d'images de profondeur.

Ensuite, nous proposons en Section 4.4, un processus de segmentation automatique qui ne requiert pour seule interaction que la sélection d'une zone quadrangulaire. Cette segmentation est cependant imparfaite, mais nous permet d'obtenir des annotations que nous pouvons utiliser pour superviser l'entraînement d'un réseau de neurones.

Ainsi, nous nous servons de ces annotations pour entraîner un réseau de neurones (Section 4.5). En effet, nous espérons que sa capacité de généralisation permettra d'améliorer les resultats obtenus précédemment. Cependant, un apprentissage direct avec la totalité des annotations détériore la qualité de la segmentation. Nous introduisons donc un processus de sélection, basé sur le coefficient de Jaccard entre l'annotation et la segmentation prédite par le modèle, afin de ne conserver que les étiquettes de bonne qualité et utilisables pour l'entraînement du réseau de neurones.

La Section 4.6 nous permet de montrer que le processus que nous proposons permet d'améliorer de manière significative la perfomance du réseau de neurones. La qualité de la segmentation finale s'en trouve donc augmentée, aussi bien lorsqu'on la compare à celle obtenue suite à un apprentissage sans sélection que par rapport à la première segmentation automatique.

Chapitre 5

Estimation de posture

Sommaire

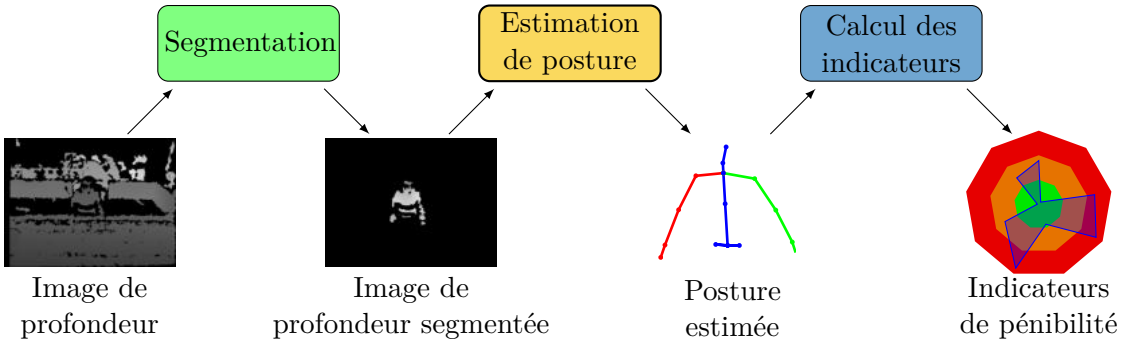


FIGURE 5.1 – Notre pipeline pour le calcul des indicateurs de pénibilité à partir des images de profondeur. Dans ce chapitre, nous considérons le premier module terminé (en vert), et nous nous concentrons sur le second module (en jaune), qui nous permettra de construire une estimation de la posture de l'opérateur humain à partir de l'image de profondeur segmentée.

Dans ce chapitre, nous nous concentrons sur le second module de notre pipeline, présenté en Figure 5.1. Nous souhaitons donc élaborer un modèle nous permettant de réaliser, de manière totalement automatisée, l'estimation de la posture de l'opérateur humain sur une image de profondeur segmentée.

Comme nous ne possédons pas d'images réelles annotées, nous débutons en construisant un ensemble d'images de profondeur de synthèse (voir Section 5.1). Ces images de profondeur de synthèse sont associées à une annotation de posture donnée sous la forme d'un squelette cinématique.

Nous réalisons ensuite une étude sur l'architecture du modèle dans le but d'optimiser la qualité de l'estimation de posture, présentée en Section 5.3. Cette étude analyse l'impact de la structure du réseau de neurones sur ses performances.

Après avoir évoqué les différentes architectures de réseaux de neurones de la littérature en Section 5.2, cette étude nous permet de construire un réseau de neurones optimal pour l'estimation de posture 3D sur des images de synthèse, que nous décrivons en Section 5.4. Afin d'accélérer son entraînement, nous proposons en outre une version simplifiée et plus portable de ce modèle optimal.

Nous analysons enfin l'efficacité du modèle sélectionné sur des images de profondeur de synthèse en Section 5.5, pour lesquels nous analysons la cohérence des indicateurs calculés (voir Figure 5.1, en bleu), puis sur des images de profondeur réelles en Section 5.6.

5.1 Génération d'images de synthèse

Dans notre contexte industriel, où l'activité humaine est analysée pour éviter les troubles musculo-squelettiques, nous sommes limités par plusieurs contraintes telles que la position des capteurs, des conditions d'éclairage incontrôlables, la présence de surfaces et le port de

FIGURE 5.2 – Exemple d'images de notre ensemble de données (ligne du haut) et de leur segmentation associée (ligne du bas). Pour un rappel du procédé de segmentation, se référer au Chapitre 4.

vêtements réfléchissants, et un nombre important d'objets en mouvement. Pour protéger la vie privée des opérateurs et favoriser l'acceptabilité de notre système, nous avons choisi d'acquérir les données en utilisant des caméras de profondeur. L'environnement industriel introduit de la complexité et du bruit dans les images de profondeur et les méthodes habituelles d'estimation de la posture humaine ne produisent pas de résultats exploitables, car notre cas d'utilisation est trop différent des conditions d'entraînement ou d'utilisation habituelle de ces méthodes. Nous avons construit ce que nous désignons comme notre ensemble de données réelles en acquérant environ 60 000 images au cours de 8 sessions enregistrées en direct dans un centre de tri des déchets.

Des exemples d'images de profondeur réelles sont présentés dans la Figure 5.2. L'opérateur est segmenté à partir de l'image en utilisant l'approche proposée dans le Chapitre 4.

Les données de profondeur sont directement acquises et stockées sur 16 bits plutôt que sur de simples images PNG 8 bits, afin de capturer correctement la valeur de profondeur de chaque pixel. En effet, la précision des valeurs de profondeur est nécessaire pour pouvoir par exemple distinguer la main de l'opérateur de l'objet qu'il peut tenir. Ce type de stockage nous permet d'effectuer une meilleure normalisation de nos données avant de les transmettre à nos réseaux, alors que les images PNG nous auraient fourni un moins bon échantillonnage de la profondeur de l'opérateur.

Malheureusement, nous ne pouvons pas utiliser directement ces données pour entraîner un réseau de neurones pour l'estimation de la posture humaine. En effet, l'apprentissage machine implique souvent un besoin critique de grandes quantités de données étiquetées pendant la phase d'entraînement. Dans notre cas, l'annotation d'une posture humaine réalisée uniquement à partir d'une image de profondeur est une tâche très difficile et très longue, en réalité presque impossible à réaliser avec une précision suffisante, car il s'agit de fournir une information en 3D sur un espace en 2D. En outre, nos contraintes industrielles

rendent probable que les conditions d'acquisition diffèrent d'un centre de tri à l'autre. La caméra de profondeur peut devoir être positionnée différemment, ou les opérateurs humains peuvent porter des vêtements réfléchissants différents, ce qui produirait un nouveau type de données, qui nécessiterait donc une nouvelle campagne d'annotation pour réentrainer le réseau. Nous pensons que ce processus n'est pas viable dans notre cas d'utilisation. C'est pourquoi nous avons décidé de nous appuyer sur un ensemble de données de synthèse étiquetées plutôt que sur des annotations faites à la main.

Le manque de données annotées nous amène donc à considérer l'utilisation d'images de synthèse pour entraîner différents algorithmes d'apprentissage. Cette approche a en effet déjà été employée dans le cadre de l'estimation de posture sur les images de profondeur par [Shotton et al., 2011] avec succès, puisque c'est ainsi que l'algorithme d'estimation de posture qui est utilisé par les capteurs Kinect a été entraîné.

5.1.1 Outils pour la génération de données de synthèse

Ces images de synthèse ont été générées grâce aux logiciels gratuits et *open source* MakeHuman [1] [The MakeHuman Team, 2020] et Blender [2] [Blender Online Community, 2020].

Le modèle 3D d'agent de tri a tout d'abord été construit à l'aide du logiciel libre et gratuit MakeHuman [The MakeHuman Team, 2020]. Ce logiciel nous a permis de réaliser un modèle 3D d'humanoïde et de les équiper d'une armature rigide afin d'animer le modèle dans de futurs traitements. Nous avons choisi d'utiliser MakeHuman plutôt qu'un modèle 3D gratuit et directement disponible en ligne car ce logiciel nos donne plus de liberté et permet en outre d'habiller le modèle afin d'augmenter la ressemblance à notre cas d'utilisation. Par ailleurs, nous n'avons conservé que la partie supérieure du modèle, car les jambes de opérateurs sont occultées par la table de tri sur les images réelles. Nous avons ensuite exporté notre modèle 3D au format MHX2 (extension `.mhx2`), directement interprétable par Blender avec l'uilisation d'un greffon dédié créé par les développeurs de MakeHuman.

La génération des images de synthèse a été réalisée au moyen du logiciel Blender [Blender Online Community, 2020]. Blender est un logiciel libre et gratuit qui dispose de fonctions avancées de modélisation 3D, telles que l'animation 3D, le scripting et la composition. Grâce à ces fonctions, nous avons pu utiliser le modèle 3D de valoriste équipé d'une armature construit précédemment pour réaliser l'animation réaliste du modèle et un rendu sous forme d'image de profondeur des images. Afin d'automatiser la génération de l'ensemble d'images de synthèse, nous avons fait appel aux possibilités de scripting du logiciel, qui autorise l'implantation de script en langage Python, grâce à une API dédiée. La gamme d'angulations pour chaque articulation de notre modèle a été déterminée grâce à une étude

1. `http://www.makehumancommunity.org`
2. `https://www.blender.org`

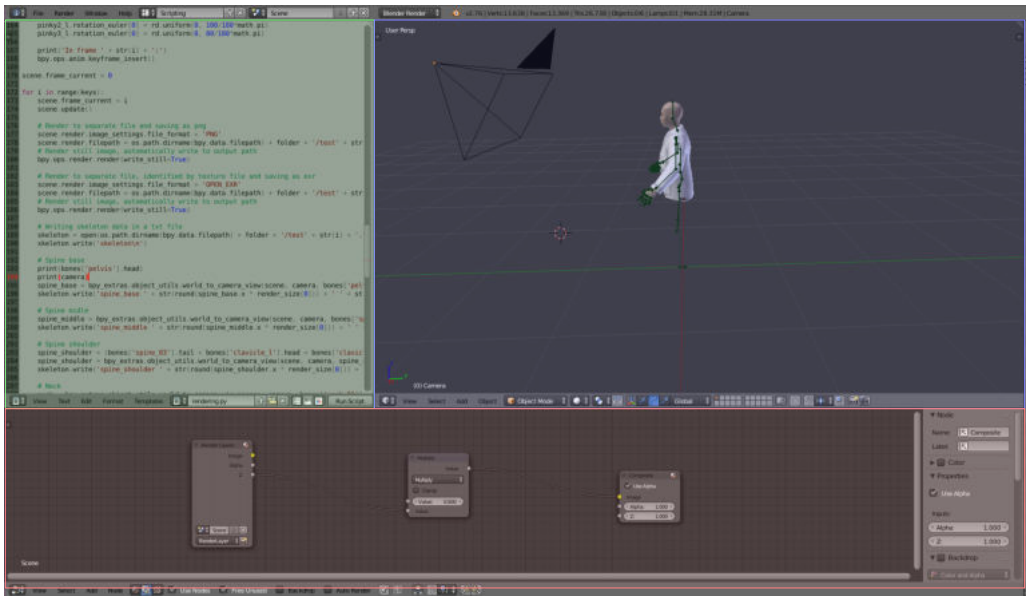


FIGURE 5.3 – Interface du logiciel Blender. On a affiché les trois principaux modules utilisés pour construire les images de profondeur de synthèse. En vert, le module de scripting, qui permet d'implanter en langage Python un script à exécuter pour construire les images de synthèse et leur annotation. En bleu, l'outil de visualisation, qui affiche et permet de déplacer la caméra et le modèle. Enfin, le module de composition, en rouge, qui sert à définir le rendu des images de profondeur de synthèse (ici, on récupère le canal Z en millimètres).

ergonomique [Dutrieux et al., 2018], pour correspondre étroitement à celles de l'activité de tri, et donc, à celles de nos données réelles. Nous construisons ainsi un ensemble d'images clés (en anglais *keyframes*) contenant des postures clés, et la transition d'une posture à l'autre est directement calculée par le logiciel, afin d'obtenir une transition fluide, si l'on souhaite utiliser des séquences d'images.

Nous pouvons aussi régler la position de la caméra. Pour que le champ de vision de celle-ci corresponde à celui de nos images de profondeur réelles, nous l'avons placée en hauteur, avec une vue plongeante sur l'opérateur. La caméra se trouve finalement à environ 3m40 de celui-ci.

La Figure 5.3 présente l'interface du logiciel Blender pour notre génération de données.

5.1.2 Format et rendu des données de synthèse

Le rendu sous forme d'image de profondeur est effectué en utilisant la distance du modèle par rapport à la caméra, c'est-à-dire en récupérant le canal Z (ou *Z-buffer*) : la valeur de chaque pixel est égale à sa distance par rapport au point caméra (voir Figure 5.4). Cette distance est mesurée en millimètres. Il est important de noter que les images finales sont exportées au format OpenEXR (extension `.exr`) plutôt qu'un autre format (PNG, JPEG,

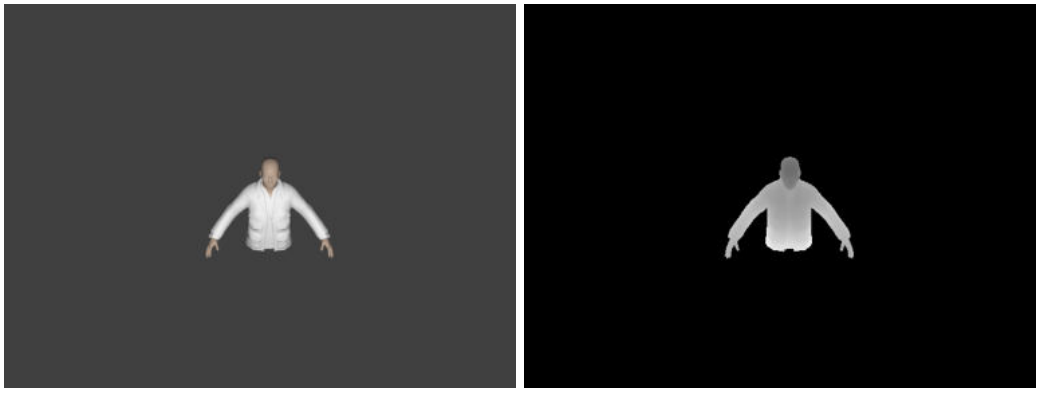


FIGURE 5.4 – Exemple de modèle avec rendu couleur (à gauche) et son canal Z (à droite). Le canal Z d'un modèle correspond à la distance de chaque pixel à la caméra, c'est donc sa profondeur. Pour le rendu du canal Z, plus la distance est importante, plus le pixel est clair, et un pixel noir signifie une absence d'information.

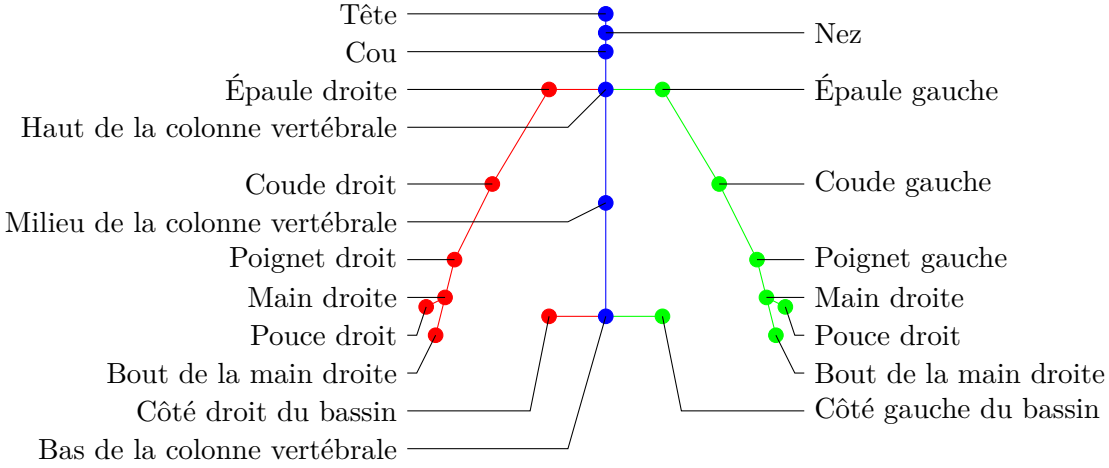


FIGURE 5.5 – Les vingt articulations utilisées pour définir la posture 3D du haut du corps.

etc.), car OpenEXR a l'avantage de pouvoir encoder les valeurs des pixels en nombre flottant sur 16 bits plutôt que d'effectuer un échantillonnage sur 8 bits comme le fait par exemple le format PNG. Ainsi, nos images de synthèse conservent le même degré de précision que les images de profondeur réelles. Aussi, à chaque image de synthèse est associée une annotation sous forme d'un fichier texte contenant les coordonnées des différentes articulations dont on souhaite estimer la position, qui définissent notre posture 3D du haut du corps. Elles ont la même résolution que les images de profondeur acquises grâce au capteur Orbbec Astra, c'est-à-dire 640×480.

La posture 3D du haut du corps est ainsi définie par les vingt articulations suivantes, illustrées en Figure 5.5 :

— *Bas de la colonne vertébrale* ;
— *Milieu de la colonne vertébrale* ;
— *Haut de la colonne vertébrale* ;
— *Cou* ;
— *Tête* ;
— *Nez* ;
— *Épaule gauche* ;
— *Épaule droite* ;
— *Coude gauche* ;
— *Coude droit* ;
— *Poignet gauche* ;
— *Poignet droit* ;
— *Main gauche* ;
— *Main droite* ;
— *Pouce gauche* ;
— *Pouce droit* ;
— *Extrémité de la main gauche* ;
— *Extrémité de la main droite* ;
— *Bassin gauche* ;
— *Bassin droit.*

La position 3D des articulations peut être définie de deux manières différentes :
— dans un premier cas, la position (u, v) du pixel contenant l'articulation correspond aux deux premières coordonnées, et la troisième représente la distance d entre le point caméra et l'articulation (en millimètres). On parle ici de *coordonnées pixelliques* ;
— dans un second cas, la position (x, y, z) correspond à une position dans l'espace 3D, et est définie par rapport à un point central, qui est le bas de la colonne vertébrale. On parle ici de *coordonnées réelles.*

Il s'avère que, pour calculer les angulations des différentes articulations, le deuxième type de coordonnées est plus intéressant. En effet, toutes les coordonnées évoluent dans le même espace, et il n'y a pas besoin d'utiliser les paramètres intrinsèques de la caméra, directement donnés ou obtenables par calibrage, pour projeter les pixels vers l'espace 3D. Pour cette raison, nous utilisons donc dans la suite les *coordonnées réelles* (x, y, z).

Des exemples d'images de profondeur générées sont donnés en Figures 5.6 et 5.7, et une image de profondeur et ses deux types d'annotations associées sont présentés en Figure 5.8.

5.1.3 Étude de la variabilité des postures à estimer

Dans cette partie, on étudie la variabilité des positions des articulations à estimer. Pour calculer la variabilité de la position des articulations, illustrée en Figure 5.9 pour les articulations du bassin, de la tête, des épaules et des poignets, on mesure leur écart-type pour chacune des coordonnées de leur position (x, y, z). Les valeurs calculées sont données en Tableau 5.1.

On observe que plus une articulation est éloignée du point de référence, plus sa variabilité est importante. En effet, ce sont bien les bras qui ont les mouvements avec la plus grande amplitude, il semble donc logique que ce soit leurs articulations qui possèdent le plus grand écart-type dans leur position, en particulier pour leurs extrémités, tels que les poignets et les mains. Nous verrons par la suite que les articulations les plus variables sont aussi les articulations dont la position est la plus difficile à estimer pour nos modèles.

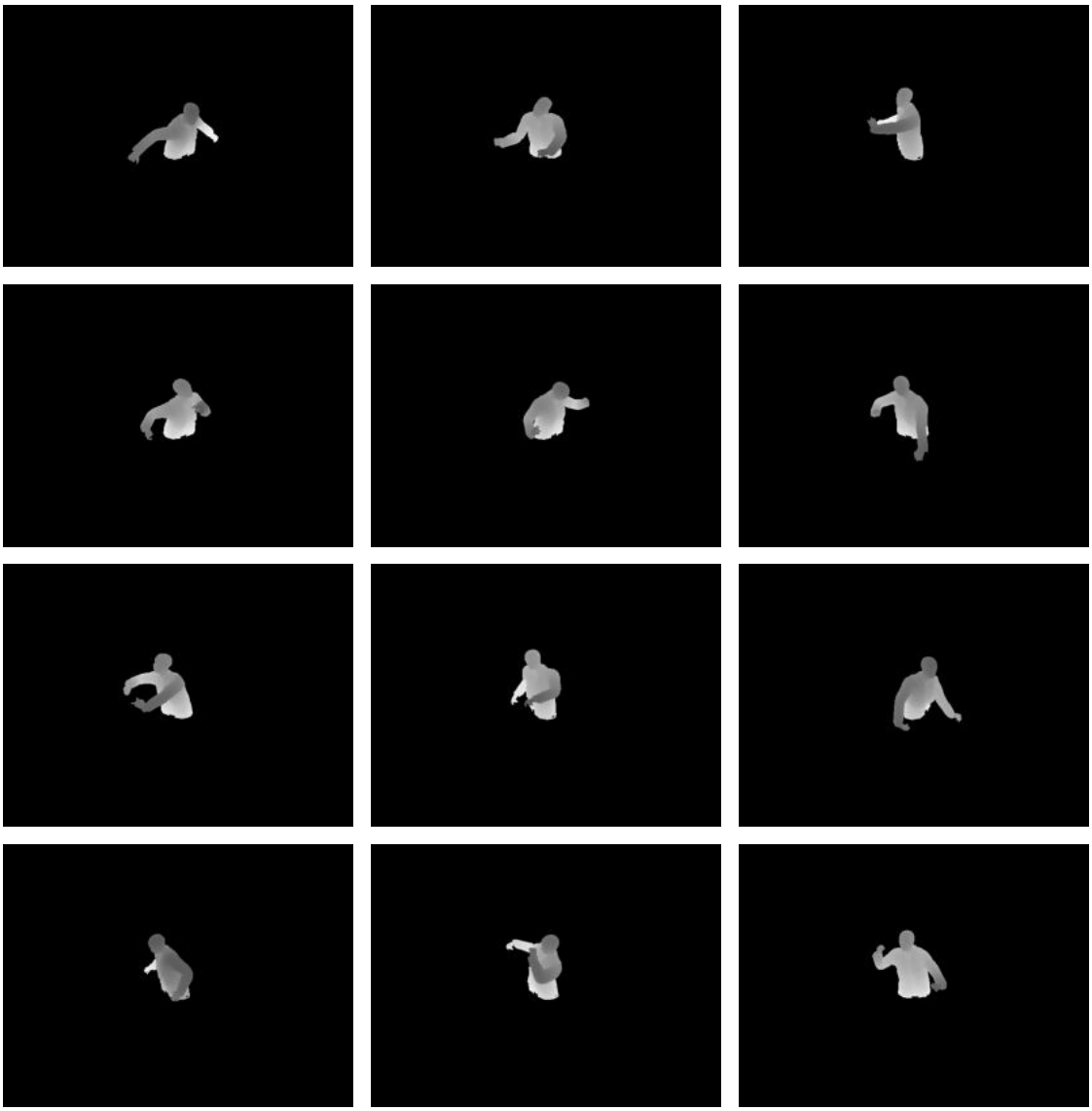


FIGURE 5.6 – Exemple d'images de profondeur de synthèse construites à l'aide de Blender et MakeHuman. Notre ensemble de données contient 200 000 images annotées décrivant une grande quantité de postures du haut du corps. Plus la profondeur est importante, plus le pixel est clair, et un pixel noir signifie une absence d'information.

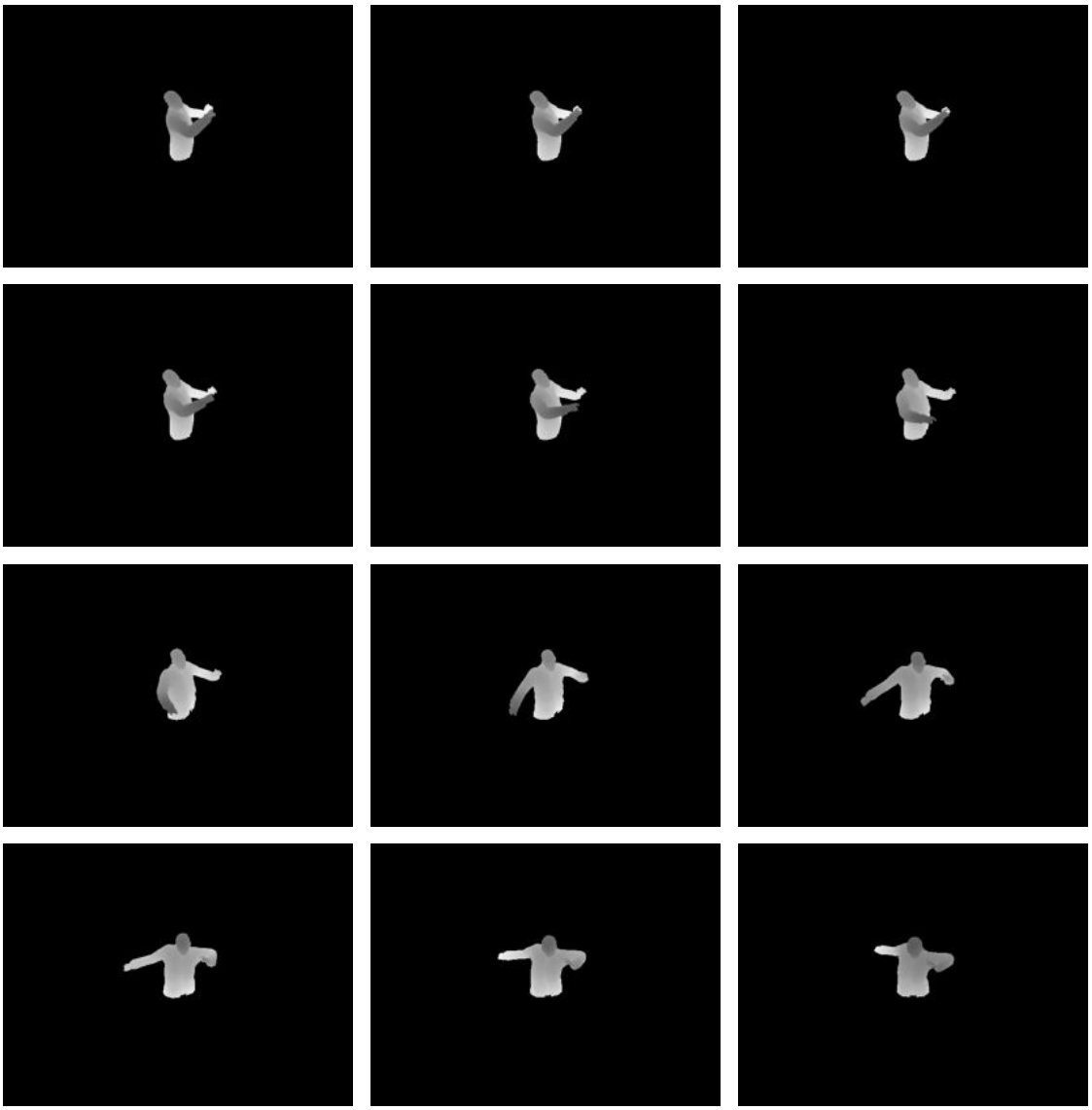
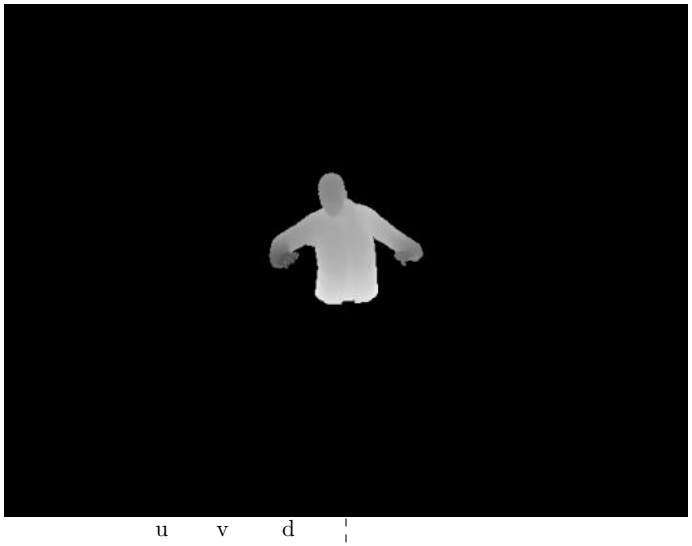


FIGURE 5.7 – Exemple de séquence de plusieurs images de profondeur de synthèse consécutives. La lecture se fait de gauche à droite, du haut vers le bas.

5.2 Estimation de posture par apprentissage profond

Cette partie est focalisée sur les approches de l'état de l'art qui utilisent l'apprentissage profond et les réseaux de neurones pour réaliser l'estimation de posture. Ces méthodes fonctionnent de la même manière que celles basées sur l'apprentissage, auxquelles elles ont succédé.

Comme pour la segmentation, si les réseaux de neurones ont été popularisés par l'utilisation d'AlexNet en 2012 [Krizhevsky et al., 2012] dans le contexte de la classification



	u	v	d
Bas de la colonne vertébrale	321	262	3142
Milieu de la colonne vertébrale	319	233	3037
Haut de la colonne vertébrale	315	203	2898
Cou	314	192	2858
Tête	308	182	2769
Nez	306	191	2688
Épaule gauche	339	200	2944
Coude gauche	368	221	2850
Poignet gauche	382	233	2635
Main gauche	375	233	2587
Pouce gauche	375	239	2558
Extrémité de la main gauche	375	244	2577
Épaule droite	290	207	2833
Coude droit	258	227	2749
Poignet droit	260	238	2525
Main droite	261	236	2460
Pouce droit	273	239	2468
Extrémité de la main droite	266	244	2445
Bassin gauche	335	261	3160
Bassin droit	306	263	3146

Coordonnées pixelliques

	x	y	z
Bas de la colonne vertébrale	0	0	0
Milieu de la colonne vertébrale	-10	220	26
Haut de la colonne vertébrale	-35	459	24
Cou	-44	538	30
Tête	-76	632	-12
Nez	-70	544	45
Épaule gauche	124	451	74
Coude gauche	300	392	-82
Poignet gauche	363	446	-306
Main gauche	315	477	-343
Pouce gauche	313	461	-390
Extrémité de la main gauche	315	426	-391
Épaule droite	-195	473	-46
Coude droit	-382	416	-189
Poignet droit	-342	485	-413
Main droite	-327	533	-459
Pouce droit	-260	517	-461
Extrémité de la main droite	-296	507	-495
Bassin gauche	100	-8	17
Bassin droit	-102	-8	-1

Coordonnées réelles

FIGURE 5.8 – Exemple d'image de profondeur de synthèse (en haut) et ses deux types d'annotations associées (en bas) : coordonnées pixelliques (u, v, d) (à gauche) et coordonnées réelles (x, y, z) (à droite). Nous utilisons vingt points pour définir la posture 3D du haut du corps de l'opérateur.

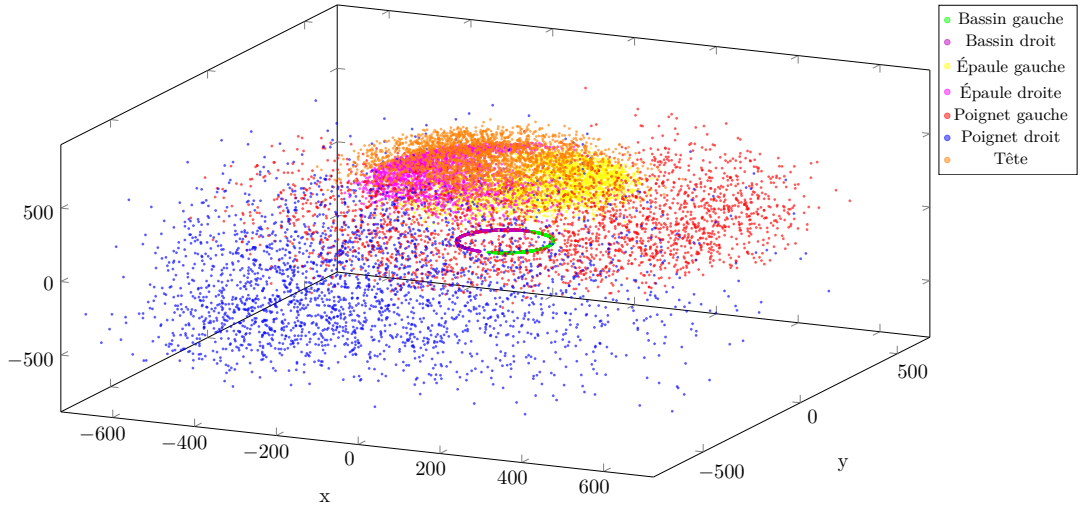


FIGURE 5.9 – Variabilité de la position de différentes articulations de notre squelette virtuel. En vert, jaune et rouge, les positions du bassin, de l'épaule et du poignet gauches, en violet, rose et bleu celles du bassin, de l'épaule et du poignet droits. En orange, les positions de la tête. On observe que certaines articulations (poignets en particulier) ont des positions beaucoup plus variables que d'autres (bassin, par exemple).

Articulation	Écart-type (mm)		
	x	y	z
Bas de la colonne vertébrale	0	0	0
Milieu de la colonne vertébrale	16.4	5.5	18.2
Haut de la colonne vertébrale	74.9	21.2	65.5
Cou	93.3	25.4	81.7
Tête	133.2	41.1	104.6
Épaule gauche	97.2	41.3	128.6
Coude gauche	164.0	94.2	221.7
Poignet gauche	257.9	161.7	265.1
Main gauche	283.6	182.2	275.9
Épaule droite	100.7	40.7	127.6
Coude droit	177.9	97.4	238.8
Poignet droit	265.5	155.5	274.0
Main droite	293.4	175.7	288.2
Bassin gauche	28.9	0.0	64.2
Bassin droit	28.4	0.0	64.4

TABLE 5.1 – Écart-type des coordonnées des articulations à estimer. On observe que plus on s'éloigne du point de référence (le bas de la colonne vertébrale), plus l'écart-type des coordonnées des articulations augmente, avec un maximum atteint au niveau des mains.

d'images, il a fallu attendre 2014 pour que les premières approches ayant recours à des réseaux de neurones profonds soient proposées [Chen and Yuille, 2014, Toshev and Szegedy, 2014]. Les réseaux de neurones utilisés sont généralement des réseaux de neurones convolutifs, car ils analysent le plus souvent des images et proposent en sortie un vecteur représentant la posture (voir Section 3.4.2). Ce vecteur peut être de taille variable en fonction du nombre de paramètres à estimer pour caractériser la posture.

Il ne faut cependant pas oublier les quelques approches qui s'étaient déjà essayées à l'estimation de posture par réseaux de neurones au début des années 2000 [Rosales and Sclaroff, 2000, Rosales and Sclaroff, 2002]. Les réseaux utilisés sont alors des perceptrons multi-couches avec une unique couche cachée, et ont pour but d'estimer la posture à partir de caractéristiques déjà extraites de l'image. Ceci est donc bien différent des approches proposées dans cette partie, où l'estimation de posture est réalisée par un réseau de neurones à partir de l'image "brute" donnée en entrée.

Nous divisons la présentation des approches d'estimation de la posture par réseau de neurones en trois catégories : tout d'abord, nous présentons les méthodes pour l'estimation de posture en deux dimensions, puis celles pour l'estimation de posture en trois dimensions. Enfin, nous présentons les approches spécifiques à l'estimation de posture sur des images de profondeur.

5.2.1 Estimation de posture en 2D

Contrairement aux approches traditionnelles, souvent orientées sur le suivi de la posture, une majorité des travaux d'estimation de la posture cherchent à estimer un posture en deux dimensions. En effet, comme ce sont des approches basées sur l'apprentissage, elles nécessitent l'utilisation d'un ou plusieurs ensembles de données étiquetées, et les postures en deux dimensions sont plus simples à annoter et donc moins coûteuses à obtenir. En général, les méthodes d'apprentissage profond pour l'estimation de la posture vont chercher à estimer la position de points d'intérêt, qui sont des articulations (comme les épaules, les coudes, les poignets, etc.) ou qui représentent des parties du corps (les mains, la tête, etc.) sur l'image par régression. En reliant ces points, on obtient une posture sous la forme d'un squelette.

Les premiers travaux portant sur l'estimation de posture en 2D par réseau de neurones ont été présentés en 2014. Dès le départ, ces approches ont pu être classées en deux catégories (voir Figure 5.10) :

— les approches basées sur une régression directe de la position des points d'intérêt : le modèle va directement estimer la position des articulations, sans passer par une étape de construction de cartes de chaleur explicite ;

— les approches basées sur les cartes de chaleur : le réseau de neurones construit une carte de chaleur pour chaque point d'intérêt, qui donne la probabilité qu'un point

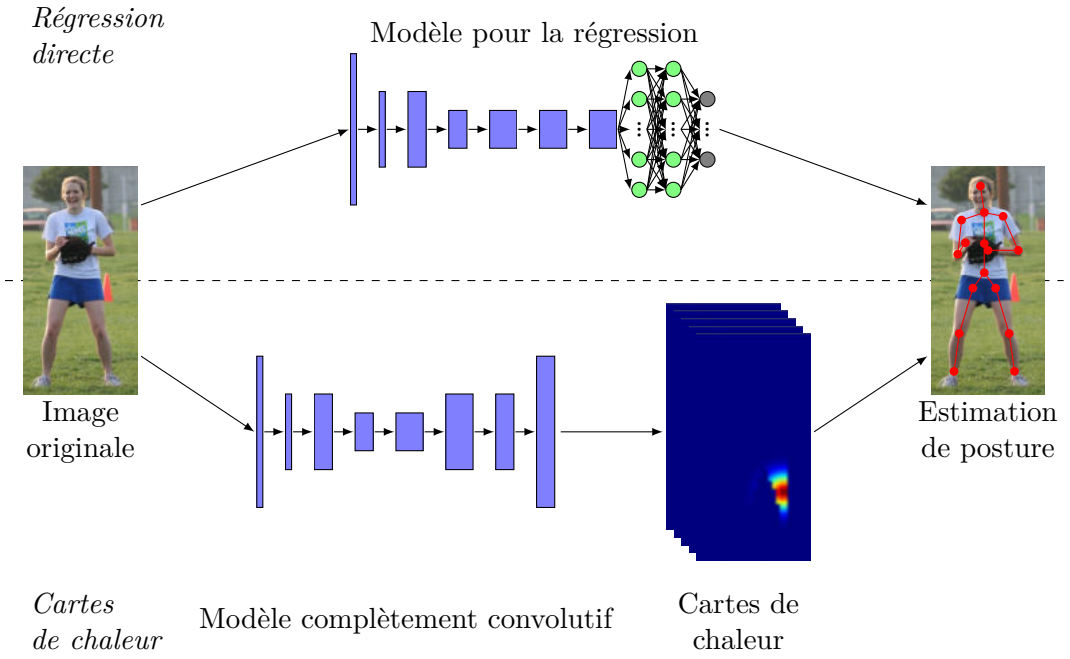


FIGURE 5.10 – Les deux types d'approches pour l'estimation de posture par réseau de neurones. En haut, l'estimation de la posture est directement réalisée par le réseau de neurones par régression. Celui-ci contient donc une partie constituée de couches totalement connectées (en vert). En bas, le réseau de neurones construit des cartes de chaleur plutôt qu'une régression directe. Les réseaux utilisés sont donc généralement des réseaux complétement convolutifs. Les cartes de chaleur sont ensuite utilisées pour déduire l'estimation de la posture.

Type d'approche	Avantages	Inconvénients
Régression directe	Rapide et direct Entraînement de bout en bout Facilement transposable au 3D	Apprentissage plus difficile Estimation de posture unique
Cartes de chaleur	Apprentissage plus simple Estimation de multiples postures	Dépendant de la résolution Coûteux en mémoire Nécessité d'un post-traitement Difficile à étendre au 3D

TABLE 5.2 – Avantages et inconvénients des approches pour l'estimation de posture par réseau de neurones par régression directe et par cartes de chaleur.

d'intérêt soit situé sur chaque pixel. Ensuite, ces cartes de chaleur sont utilisées pour obtenir l'estimation finale du point d'intérêt.

Ces deux types d'approches ont leur propres avantages et inconvénients, résumés dans le Tableau 5.2. En particulier, la régression directe des coordonnées est plus difficile à apprendre : en effet, il faut estimer des coordonnées, qui sont des métadonnées de l'image. Ceci va au-delà de la détection de certaines caractéristiques, sur laquelle repose l'estimation par cartes de chaleur. Cependant, la régression directe a l'avantage d'être simple et rapide, alors que les cartes de chaleur sont coûteuses en mémoire par rapport à leur résolution, et qu'un post-traitement doit leur être appliqué pour obtenir l'estimation de posture finale. En outre, l'estimation par régression directe est plus simplement extensible au problème de l'estimation de posture en 3D, alors que les cartes de chaleur permettent d'estimer la posture de plusieurs personnes sur la même image sans passer par une détection préalable. Ainsi, il est difficile de se prononcer sur la meilleure catégorie d'approches, celle-ci dépendant de l'application visée et des ressources disponibles.

Estimation de posture par régression directe

[Toshev and Szegedy, 2014] sont les premiers à proposer un modèle de réseau de neurones profond pour l'estimation de posture, connu sous le nom de DeepPose, illustré en Figure 5.11. Ce modèle utilise plusieurs réseaux de neurones : tout d'abord, un premier réseau de neurone, appelé "régresseur" prend en entrée l'image complète à analyser et estime une première posture complète, puis la position de chaque point d'intérêt qui compose la posture est raffinée par une cascade de régressions effectuée par des réseaux de neurones, appelés raffineurs, qui travaillent sur des patchs centrés sur la position estimée du point d'intérêt considéré. On a un réseau raffineur par point d'intérêt à estimer. L'architecture de tous les réseaux de neurones est directement inspirée de celle de [Krizhevsky et al., 2012].

Les auteurs de [Chen and Yuille, 2014] proposent d'utiliser un réseau de neurones convolutif pour apprendre des probabilités conditionnelles pour la présence de parties du corps et leurs relations par paire sur des patchs d'image, par rapport à un modèle cinématique de la posture humaine. Ce modèle peut donc estimer la posture à partir de patchs, tout en modélisant les relations entre les différentes parties du corps. Cette approche a été étendue à l'estimation de posture de plusieurs personnes dans [Chen and Yuille, 2015].

Une approche décrite par [Carreira et al., 2016] fait appel à un modèle auto-correcteur : l'estimation de posture est effectuée puis raffinée itérativement par le même réseau. La solution initiale est donc progressivement modifiée par un processus appelé *iterative error feedback*. Le problème de l'estimation n'est plus vu comme une tâche de prédiction, mais plutôt comme une tâche de correction.

L'approche introduite par [Sun et al., 2017] tient compte de la structure du corps humain, en utilisant des segments représentant les os plutôt que les points d'intérêt. Cette

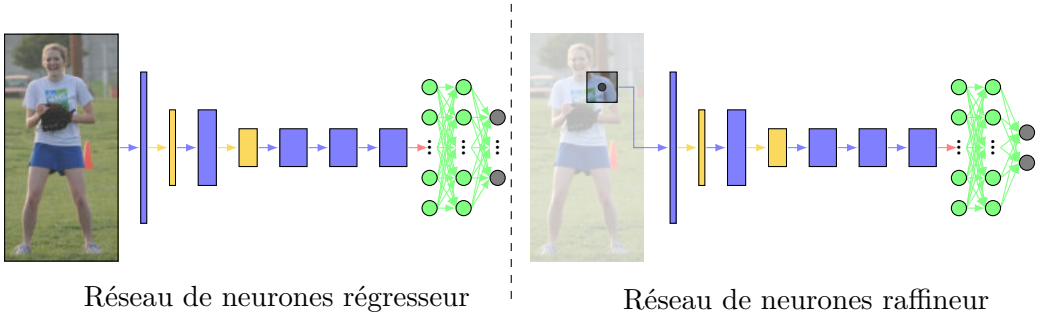


FIGURE 5.11 – Le fonctionnement du modèle DeepPose proposé par [Toshev and Szegedy, 2014]. L'image est tout d'abord analysée par un réseau de neurones qui effectue la régression de la posture complète (à gauche). Ensuite, un réseau de neurones propre à chaque point d'intérêt (à droite) est utilisé en cascade pour raffiner la position de chaque point d'intérêt. À chaque itération, ce réseau prend en entrée un patch de l'image centré sur l'estimation précédente de la position du point d'intérêt.

approche est baptisée "régression compositionnelle de la posture". Selon ses auteurs, se servir de segments rend l'apprentissage plus simple et plus stable. Pour l'apprentissage, la fonction de perte utilisée est définie à l'aide des connexions entre les segments au niveau des articulations. Cette fonction de perte encode les interactions des os dans la posture humaine. Le réseau de neurones utilisé est un ResNet-50 [He et al., 2016], avec une dernière couche adaptée au nombre de coordonnées à estimer. Cette approche a aussi été utilisée pour l'estimation de posture en 3D.

Une méthode présentée par [Luvizon et al., 2019] permet d'estimer directement la posture en passant par des cartes de chaleur implicites. La méthode propose une transformation des cartes de chaleur en coordonnées des points d'intérêt de manière totalement différenciable grâce à une fonction de *Soft-argmax*. Le modèle peut donc être entraîné de bout en bout, et donner des résultats comparables à ceux proposés par les approches utilisant les cartes de chaleur. Le réseau de neurones est composé de deux parties : la première partie du réseau a pour objectif d'extraire les caractéristiques de l'image, puis un enchaînement de K blocs de prédiction sert à raffiner ces caractéristiques pour estimer la posture. Chaque bloc de prédiction donne une estimation de posture intermédiaire qui est utilisée lors de l'apprentissage. L'estimation de posture finale est fournie par le dernier bloc.

Estimation de posture par cartes de chaleur

De nombreux travaux proposent une estimation de posture par cartes de chaleur. Une des premières approches en ce sens est proposée par [Jain et al., 2014]. Cette méthode a recours un modèle spatial vu comme un champ de Markov aléatoire (voir Section 3.1.1) avec une structure et des a priori explicitement définis. Pour chaque point d'intérêt à

estimer, un réseau de neurones est estimé. Leur architecture est basée sur celle de LeNet-5 [LeCun et al., 1998]. Cette technique a été améliorée par [Tompson et al., 2014] : les auteurs utilisent alors un modèle multi-résolution qui réalise une première estimation de la position des différents points d'intérêt grâce à la construction de cartes de chaleur. Celles-ci sont ensuite raffinées en supprimant les éléments aberrants à l'aide d'un modèle spatial appris implicitement.

[Fan et al., 2015] présente une méthode utilisant des réseaux de neurones convolutifs à deux entrées : pour effectuer l'estimation de la position d'une articulation, le réseau de neurones prend en entrée un patch d'une image contenant un point d'intérêt, un patch du corps complet associé à un masque qui donne la position du patch du point d'intérêt par rapport au patch du corps complet. Ces patchs sont construits à l'aide d'un détecteur d'objet pour les points d'intérêt ou d'un détecteur de personnes pour les corps complets. Le réseau estime la position de l'articulation en proposant une localisation sous forme de coordonnées pixelliques sur son patch et une carte de chaleur sur le patch du corps complet.

[Bulat and Tzimiropoulos, 2016] introduisent une technique pour l'estimation de posture composée de deux modules : un premier module effectue une détection des parties du corps sous la forme de cartes de chaleur, puis le second module effectue une régression de ces cartes de chaleur pour les affiner. Selon ses auteurs, ce modèle est efficace car il permet au second module de se concentrer sur la zone où se situe la partie du corps, mais aussi parce qu'il gère les occultations : la carte de chaleur associée à un point d'intérêt occulté reflète une faible confiance de la part du module de détection, et indique au second module d'utiliser l'information contextuelle.

De nombreux modèles cherchent à affiner de manière successive les cartes de chaleur afin d'améliorer leur qualité. Les *Convolutional Pose Machines* ont été proposées par [Wei et al., 2016], et s'inspirent directement des *Pose Machines* introduites par [Ramakrishna et al., 2014], pour l'apprentissage implicite des dépendances entre les paramètres pour l'estimation de posture. Elles réalisent l'estimation de posture 2D par utilisation séquentielle de réseaux de neurones qui prennent en entrée les cartes de chaleur prédites à l'itération précédente en plus de l'image d'entrée (voir Figure 5.12) : les cartes de chaleur sont ainsi raffinées itérativement, et les cartes de chaleur intermédiaires sont utiles à l'apprentissage, pour éviter le phénomène de disparition du gradient. [Newell et al., 2016] proposent une nouvelle architecture de réseau de neurones convolutif. Cette architecture est appelée "*stacked hourglass*" (traduisible par "sabliers empilés"), qui enchaîne les modules en forme de sablier (voir Figure 5.13), utilisant les blocs résiduels [He et al., 2016]. Cette architecture en forme de sablier sert à traiter les caractéristiques à de multiples échelles et de les consolider pour capturer les relations spatiales des différentes parties du corps. La répétition des sabliers avec une supervision intermédiaire sert à raffiner la construction des cartes de chaleur et donc à améliorer la qualité de l'estimation de posture. En effet, chaque sablier

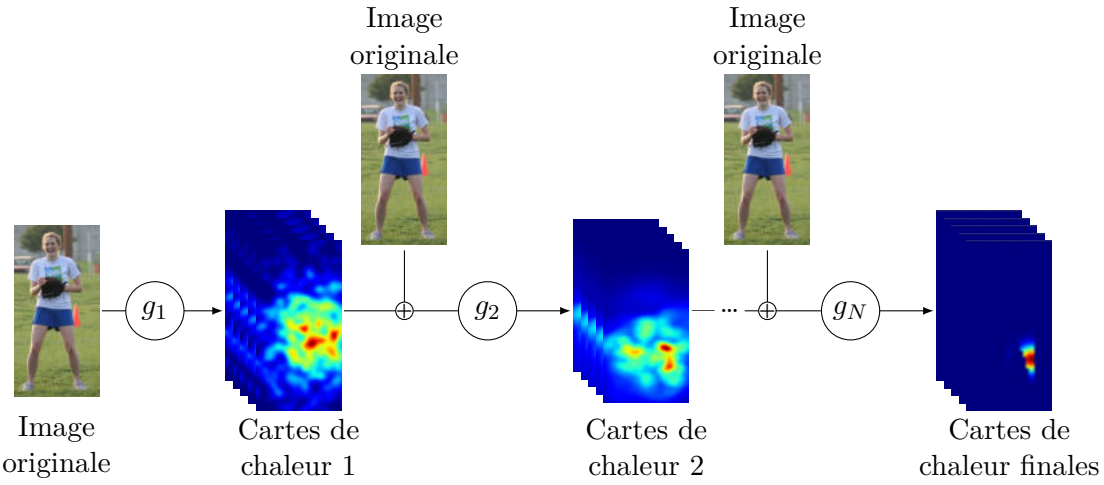


FIGURE 5.12 – Architecture du modèle des *Convolutional Pose Machines* proposées par [Wei et al., 2016]. La construction des cartes de chaleur est réalisée à l'aide d'une séquence de réseaux de neurones convolutifs. Le premier réseau de neurones g_1 ne travaille que sur l'image initiale. Les réseaux de neurones suivants g_2 à g_N utilisent l'image originale associée aux cartes de chaleur construites par les réseaux précédents.

construit des cartes de chaleur, utilisées pendant l'apprentissage. La méthode présentée par [Chu et al., 2017] reprend l'architecture *stacked hourglass* en améliorant les connexions entre les couches de même niveau, et en utilisant un champ aléatoire conditionnel pour modéliser les corrélations entre les régions voisines des cartes de chaleur. En parallèle, [Yang et al., 2017] propose d'améliorer l'architecture en utilisant un module résiduel pyramidal, pour introduire l'invariance par rapport à l'échelle. Cette méthode utilise les sabliers comme structure de base, en remplaçant la structure résiduelle des blocs par un module résiduel pyramidal. Une approche très similaire est utilisée dans [Wang et al., 2019a], pour construire un modèle du corps à plusieurs niveaux. [Belagiannis and Zisserman, 2017] introduisent un modèle de réseau de neurones utilisant un module récurrent, exécuté de manière itérative pour améliorer les résultats. Ce module permet de saisir itérativement le contexte, pour affiner la localisation, mais aussi prédire les occultations.

Certaines approches ont cherché à travailler au niveau des cartes de caractéristiques pour tenir compte des corrélations entre les positions des différents points d'intérêt. Une approche présentée par [Yang et al., 2016] propose de régulariser l'estimation de la posture à l'aide d'un mélange de parties déformables : lors de l'inférence finale, les points d'intérêt influencent leurs voisins pour obtenir la meilleure configuration possible. [Chu et al., 2016] propose de raisonner au niveau des caractéristiques extraites par le réseau de neurones : les relations entre les caractéristiques sont apprises à l'aide de noyaux de transformations géométriques. L'utilisation d'un arbre bi-directionnel sert à chaque point d'intérêt pour interagir avec les points qui lui sont corrélés pour optimiser ses caractéristiques et donc

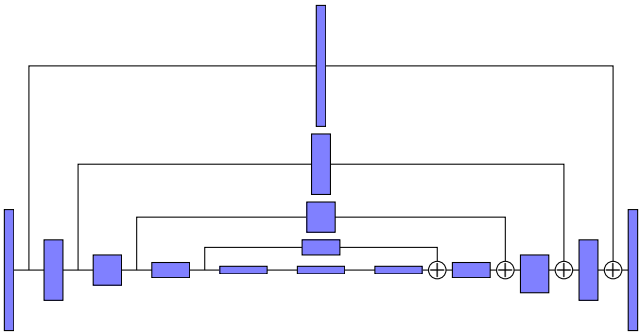


FIGURE 5.13 – La structure d'un sablier utilisé dans l'architecture "*stacked hourglass*" proposée par [Newell et al., 2016], où plusieurs de ces structures sont enchaînées. Chaque bloc (en bleu) peut être composé d'un bloc résiduel [He et al., 2016], utilisé dans l'architecture originale, ou d'une ou plusieurs couches convolutives. C'est sur la modification de la construction de ces blocs que se basent la plupart des approches utilisant les sabliers comme élément constitutif de base [Chu et al., 2017, Yang et al., 2017].

l'estimation de posture finale. [Ning et al., 2017] utilise des connaissances externes, fournies au réseau de neurones à l'aide d'une projection, apprise à l'aide d'une fonction de coût auxiliaire, de caractéristiques visuelles pour représenter les contraintes du corps humain. Les travaux de [Tang et al., 2018] cherchent eux à construire un modèle composé en utilisant des cartes de chaleur à différents niveaux, qui permettent de déduire les liens entre les différentes parties du corps.

Des méthodes récentes proposent d'estimer la posture tout en effectuant une décomposition sémantique de l'utilisateur. En effet, ces tâches étant connexes, elles peuvent s'influencer et donc s'améliorer entre elles. [Nie et al., 2018] propose par exemple d'estimer la posture suite à une segmentation du corps humain en ses différentes parties. [Liang et al., 2018] présente un modèle qui réalise conjointement la décomposition en partie et l'estimation de la posture, pour tirer partie de la corrélation entre les deux problèmes. Ces approches sont toutefois difficile à mettre en place en pratique, car elles demandent des annotations supplémentaires concernant les parties du corps, en plus de la position des points d'intérêt à estimer.

Pour corriger les erreurs liées aux occultations, d'autres approches ont recours aux réseaux antagonistes génératifs. Par exemple, [Chen et al., 2017c] se sert d'un discriminateur pour distinguer les vraies postures des postures estimées, qui sont parfois structurellement aberrantes. D'après les auteurs, l'utilisation du discriminateur permet au générateur de mieux apprendre les contraintes sur la structure du corps humain, et donc à mieux estimer la posture, en particulier lorsqu'il y a des auto-occultations.

Des approches essaient d'estimer la posture sur des séquences vidéos. [Pfister et al., 2015] proposent un réseau de neurones qui prend en entrée une suite d'images consécutives, et construit des cartes de chaleur propres à chaque image. Ces cartes de chaleurs sont ensuite

régularisées à l'aide du flux optique et mélangées en une carte de chaleur composite, constituée d'un mélange des cartes de chaleur des différentes images effectué à l'aide d'une couche de convolution. Cette dernière carte de chaleur est utilisée pour extraire la position du point d'intérêt en prenant son maximum global.

Estimation de la posture de plusieurs personnes

Les réseaux de neurones ont aussi été utilisés pour effectuer l'estimation de posture 2D de plusieurs personnes sur la même image. Comme pour les méthodes traditionnelles, on distingue les approches *top-down* et *bottom-up* : les approches *top down* effectuent l'estimation de posture d'une seule personne après détection sur l'image, alors que les approches *bottom-up* repèrent d'abord les parties du corps, qu'elles tentent alors d'assembler pour former un tout cohérent.

Une des premières approches *top-down* utilisant les réseaux de neurones est proposée par [Iqbal and Gall, 2016]. Chaque individu est tout d'abord détecté à l'aide d'un détecteur d'humain [Ren et al., 2015], puis des positions candidates pour chaque articulation sont calculées à l'aide des *Convolutional Pose Machines* [Wei et al., 2016], en se concentrant sur les points d'intérêt de la personne détectée, ceux des autres personnes étant simplement supprimés. Une approche très similaire est décrite par [Papandreou et al., 2017] : le même détecteur de personnes est utilisé, et la différence se fait au niveau de l'estimation de posture individuelle. Ici, c'est un ResNet-101 [He et al., 2016] qui sert à estimer la posture. Une autre méthode *top-down* a été introduite par [Fang et al., 2017] : ses auteurs remarquent que l'imprécision des détections fausse souvent l'estimation de la posture. Ils présentent donc un modèle facilitant l'estimation de posture malgré des boites englobantes imprécises. Ce modèle s'appuie sur deux réseaux de neurones qui permettent de corriger les détections extraites des images pour les faire ressembler aux images ayant servi pour l'apprentissage de l'estimateur de posture pour une seule personne. Cette approche a été étendue pour le suivi sur des séquences vidéo par [Xiu et al., 2018]. [Chen et al., 2018b] propose l'utilisation de deux réseaux de neurones en forme de pyramide pour l'estimation de la position des points d'intérêt : un premier réseau de neurones réalise une estimation globale de la position de chacun des points d'intérêt, qui est affinée par un second réseau de neurones qui se sert de l'information contextuelle pour améliorer la prédiction des articulations occultées. Enfin, une approche introduite par [Sun et al., 2019] propose de conserver une haute résolution pour améliorer les résultats de l'estimation de posture. Contrairement aux autres méthodes, dont les réseaux de neurones contiennent souvent une partie à basse résolution après plusieurs sous-échantillonnages, les auteurs proposent de conserver un traitement à haute résolution tout en traitant en parallèle des résolutions plus basses. Le détecteur de personne qui sert à séparer les individus avant l'estimation de posture est celui de [Xiao et al., 2018].

Les méthodes les plus efficaces sont cependant les approches *bottom-up*, car elles sont

moins dépendantes de la détection préalable des individus. La première approche de ce type a été baptisée DeepCut [Pishchulin et al., 2016]. L'estimation de posture est réalisée en résolvant un problème de coupes à coût minimal dans le graphe des connexions entre les articulations estimées (voir Section 3.1.2) : les points d'intérêts sont les sommets du graphe, et les relations entre les points sont symbolisées par ses arêtes, et la force de leur connexion représente la probabilité que les deux points appartiennent à la même personne. Cette méthode a été améliorée par DeeperCut [Insafutdinov et al., 2016], qui propose trois modifications : le réseau utilisé est plus profond, pour construire des cartes de chaleur plus précises, ce même réseau facilite la construction des relations en raisonnant sur la configuration de la posture, et le procédé de coupe dans le graphe est simplifié et accéléré. Cependant, cette étape est toujours coûteuse en temps de calcul. La méthode a donc de nouveau été améliorée par [Insafutdinov et al., 2017], en simplifiant le graphe des relations entre les articulations et en réalisant la reconstruction des postures par un réseau de neurones. La méthode la plus populaire a été proposée par [Cao et al., 2017]. Elle a recours à une représentation non paramétrique appelée *Part Affinity Fields* (traduisible par "champs d'affinité des parties"). Cette représentation symbolise les liens entre les différents points d'intérêt détectés. La reconstruction des postures complètes est réalisée par un algorithme glouton qui combine les *Part Affinity Fields* et les cartes de chaleur pour chacune des articulations. Cette approche est aussi connue sous le nom d'OpenPose[3] [Cao et al., 2018]. [Newell et al., 2017] présente une approche qui se sert d'une intégration associative pour superviser la tâche de détection des points d'intérêt et leur regroupement. Cette méthode construit simultanément les cartes de chaleur ainsi qu'un marquage pour chaque foyer. Le marquage associé à des foyers correspondant à des articulations différentes sont similaires si ils appartiennent à la même personne et éloignés lorsqu'ils font partie de postures différentes. MultiPoseNet [Kocabas et al., 2018] est une autre méthode *bottom-up*. Les auteurs introduisent un modèle qui réalise à la fois la détection des personnes, leur segmentation et l'estimation de leur posture. Le modèle proposé est divisible en sous réseaux de détection et de calcul de le position des articulations, dont les sorties sont combinées comme entrée d'un réseau final pour reconstituer les postures complètes. Une approche proposée par [Papandreou et al., 2018] consiste en un réseau de neurones convolutif qui construit différents types de caractéristiques, dont des cartes de chaleur et des cartes de décalages à différentes échelles, associées ensuite lors d'une transformée de Hough [Hough, 1962], dont le résultat est utilisé par un décodeur de poses basé sur un algorithme glouton.

5.2.2 Estimation de posture en 3D

Certains travaux d'estimation de la posture utilisant les réseaux de neurones cherchent à estimer une posture en trois dimensions. On peut distinguer deux grands types d'ap-

3. https://github.com/CMU-Perceptual-Computing-Lab/openpose

proches :

— les approches basées sur le transfert ou la reconstruction : ces méthodes reposent sur la prédiction 2D de la posture, qu'elles transfèrent dans le domaine 3D. On parle en anglais d'approches de *lifting* ;

— les approches directes : comme en deux dimensions, le modèle va directement estimer la position des articulations, en 3D cette fois-ci, ou bien à construire des cartes de chaleur volumétriques, composées de voxels.

Enfin, certaines approches ont cherché à estimer simultanément la posture 2D et la posture 3D.

Un important obstacle à ces travaux est la rareté des données annotées. En effet, la plupart des ensemble de données disponibles ont été acquises dans le cadre de campagnes de *Motion Capture* [Sigal et al., 2010, Ionescu et al., 2013, Mehta et al., 2017a]. La solution la plus directe est l'augmentation de données, réalisée par construction de mosaïques de postures par [Rogez and Schmid, 2016]. Cette approche permet de générer un large ensemble d'images synthétiques réalistes par combinaison de morceaux d'images, dont la posture 2D correspond à la projection de la posture 3D candidate. [Chen et al., 2016b] propose d'utiliser un modèle de synthèse en 3D, dont on connaît la posture en trois dimensions, projeté sur un arrière plan réel pour entraîner un réseau de neurones. Une autre approche cherche à ajouter une information faible sur la profondeur relative des points clés, en les ordonnant par rapport à cette troisième coordonnée, mais sans donner une valeur précise [Pavlakos et al., 2018]. Enfin, des travaux récemment proposés se servent de l'adaptation de domaine pour estimer une posture en 3D sur des images couleur à partir de données synthétiques de profondeur [Zhang et al., 2019].

Estimation de posture en 3D depuis le 2D

Les approches les plus nombreuses cherchent à déduire la posture 3D depuis une estimation de la posture 2D. En particulier, certaines approches n'utilisent un réseau de neurones que pour construire la posture en deux dimensions. C'est le cas de SMPL [Bogo et al., 2016], qui se base sur une estimation de posture 2D réalisée par DeepCut [Pishchulin et al., 2016], pour ensuite y superposer un modèle en trois dimensions, dont le squelette 3D est projeté sur la posture estimée en 2D. L'objectif est alors de minimiser l'erreur entre les deux squelettes. Une méthode proposée par [Chen and Ramanan, 2017] effectue tout d'abord une estimation de posture en deux dimensions par *Convolutional Pose Machines* [Wei et al., 2016], puis trouve la posture 3D qui correspond le mieux parmi une librairie de postures 3D. Une approche similaire est étudiée par [Zhou et al., 2016b] : à partir de la posture 2D estimée par un réseau de neurones, la posture 3D est extraite d'un dictionnaire.

[Tome et al., 2017] présente une technique où la posture en deux dimensions et la posture en trois dimensions interagissent pour s'améliorer l'une et l'autre. Le réseau de neurones est divisé en plusieurs parties, dont chacune des parties produit une estimation de posture

intermédiaire en deux dimensions, qui sert pour l'apprentissage, mais contient une couche qui construit une estimation de posture 3D puis la reprojette sur des cartes de chaleur en deux dimensions. La sortie finale du réseau de neurones est une posture 2D, à laquelle on applique un modèle probabiliste de posture 3D pour estimer la posture 3D finale.

Par la suite, des approches ont cherché à se servir de l'apprentissage profond pour estimer la posture 3D à partir de la posture 2D. [Mehta et al., 2017a] le fait à l'aide d'apprentissage par transfert : les caractéristiques utiles apprises par un réseau de neurones dans le cadre de l'estimation de posture en 2D sont réutilisées pour effectuer l'estimation de posture 3D. La méthode proposée par [Nie et al., 2017] se sert de l'estimation de posture effectuée en deux dimensions et de patchs correspondant aux parties du corps localisées par l'estimation de posture 2D pour ajouter une troisième coordonnée à l'estimation de posture en 2D. L'approche de [Tekin et al., 2017] divise le réseau en deux flux : un premier flux va traiter l'information pour construire des cartes de chaleur pour l'estimation de posture en 2D, alors qu'un second flux va chercher des caractéristiques en trois dimensions. La fusion de ces deux flux permet d'aboutir à l'estimation de posture en 3D finale. Une autre méthode introduite par [Moreno-Noguer, 2017] utilise la régression vers une matrice de distance entre les points d'intérêt en trois dimensions depuis la matrice des distances en deux dimensions par un réseau de neurones convolutif. Une approche proposée par [Zhou et al., 2017] mélange des ensembles de données pour l'estimation de posture en 2D et en 3D, pour apprendre à estimer conjointement la posture en 2D et en 3D : un premier sous-réseau calcule des caractéristiques, puis un module basé sur les *stacked hourglass* effectue l'estimation de posture en 2D. Un module final combine ces estimations de posture en deux dimensions aux caractéristiques calculées au début du réseau pour régresser la troisième coordonnée de posture et obtenir une posture en 3D.

Estimation de posture 3D par approche directe

La première approche d'estimation de posture 3D par réseau de neurones et une régression directe est proposée par [Li and Chan, 2014]. Le réseau de neurones est entraîné à l'aide de deux sorties distinctes : une première effectue la détection de parties du corps, alors que la seconde propose la régression de la posture 3D. Apprendre conjointement ces deux problèmes permet de se servir de la corrélation qui existe entre eux pour améliorer l'estimation de posture.

Les auteurs de [Li et al., 2015] voient le problème d'estimation de posture 3D comme un problème de correspondance : leur approche cherche à mesurer si une image et une estimation de posture 3D se correspondent : le réseau de neurones prend donc en entrée une image et l'estimation de posture 3D associée, et donne en sortie un score de correspondance, élevé si la posture est proche de celle présente sur l'image. Le modèle est cependant en mesure de proposer lui aussi une posture 3D associée à l'image.

Les travaux de [Tekin et al., 2016] décrivent une méthode qui a recours à la représenta-

tion latente de la posture 3D construite grâce à un auto-encodeur pour estimer la posture 3D sur une image. Ainsi, un auto-encodeur est tout d'abord entraîné sur les postures en trois dimensions pour construire un espace latent des postures. Ensuite, un encodeur est entraîné pour projeter les images dans l'espace latent des postures. Le décodeur permet enfin d'estimer la posture associée à l'image en la reconstruisant à partir de l'information latente.

Une approche présentée par [Zhou et al., 2016a] utilise un modèle cinématique directement intégré dans le modèle pour imposer des contraintes structurelles aux estimations de postures 3D. Ce modèle est utilisé pour définir la fonction de coût qui sert à l'apprentissage du réseau de neurones, dans une couche finale spécifique appelée "couche cinématique".

[Pavlakos et al., 2017] propose l'utilisation de cartes de chaleur volumétriques pour estimer la posture 3D à partir d'une image unique. L'estimation de la troisième coordonnée des articulations est réalisée à l'aide de plusieurs blocs complétement convolutifs en série : à chaque sortie de bloc, la résolution de la troisième dimension est augmentée pour obtenir un résultat de plus en plus précis. Les sorties de chaque bloc servent pendant l'apprentissage comme sorties intermédiaires.

Une méthode proposée par [Sun et al., 2018] propose d'utiliser une fontion de *Soft-argmax* afin d'effectuer la régression directe d'une posture en trois dimensions depuis des cartes de chaleurs générées implicitement par le réseau de neurones. Ceci permet d'entraîner le modèle de bout en bout.

Une approche décrite dans [Pavlakos et al., 2018] a recours à un apprentissage faiblement supervisé au niveau de la troisième coordonnée pour effectuer l'estimation de posture en 3D : au moment de l'apprentissage, on connaît seulement les relations de profondeur des différents points d'intérêt (plus proche que, plus éloigné que). Ceci rend possible l'utilisation des ensembles de données où seules les postures 2D sont annotées pour entraîner un modèle qui estime la posture en 3D.

Estimation simultanée des postures 2D et 3D

Certaines approches ont finalement cherché à estimer en même temps la posture en 2D et en 3D. Le LCR-Net (pour *Localisation*, *Classification*, *Régression*), proposé par [Rogez et al., 2017] puis amélioré dans [Rogez et al., 2019] permet de construire simultanément des estimations de posture 2D et 3D, pour plusieurs personnes sur la même image. Comme son nom l'indique, ce modèle de réseau de neurones est constitué de trois éléments :

— un module de localisation, qui détecte les individus et propose des postures ;
— un composant de classification, qui donne un score à chacune des postures proposées ;
— un élement de régression, qui affine la posture proposée pour chaque individu.

[Mehta et al., 2017b] introduit, selon ses auteurs, la première méthode qui permet d'estimer la posture complète d'un individu de manière stable, cohérente dans le temps et en temps réel. Cette approche utilise un réseau de neurones convolutif qui régresse conjoin-

tement les postures 2D et 3D. Un second composant combine ces deux estimations pour reconstruire la posture 3D stable et complète.

[Luvizon et al., 2018] propose un modèle qui réalise plusieurs tâches : il effectue l'estimation de postures en 2D et en 3D, tout en reconnaissant les actions. Ceci permet d'entraîner le réseau de neurones avec des ensembles de données correspondant aux différents problèmes à résoudre et de le faire ainsi apprendre pour toutes les tâches en même temps.

Les auteurs de [Mehta et al., 2018] proposent une méthode pour l'estimation conjointe de la posture 2D et 3D de personnes multiples. Des cartes de chaleur volumétriques, robustes par rapport aux occultations grâce à l'introduction de redondance, permettent de donner la position des différentes articulations, qui sont ensuite assemblées à l'aide des *Part Affinity Fields* proposés par [Cao et al., 2017].

5.2.3 Estimation de posture sur les images de profondeur

Grâce au succès et à la démocratisation du capteur Kinect, qui a mis en lumière les capteurs RGB-D et donc les images de profondeur, et malgré la fiabilité de l'estimation de posture fournie par la Kinect dans de nombreux cas d'utilisation usuels, certains travaux ont cherché à estimer la posture sur une image de profondeur à l'aide des réseaux de neurones. Les travaux pour l'estimation de la position des mains sur des images de profondeur, ne sont encore une fois pas décrits dans cette partie. Une revue récente de l'état de l'art sur ce sujet est présentée par [Li et al., 2019].

Les première approche d'estimation de posture sur les images de profondeur par apprentissage profond a été proposée par [Jiu et al., 2014]. Cette approche ne présente pas à proprement parler une estimation de posture explicite, mais plutôt une segmentation du corps en différentes parties. Cette segmentation est effectuée à l'aide d'un réseau de neurones convolutif qui travaille à plusieurs échelles pour extraire des caractéristiques, inspiré de [Farabet et al., 2012]. Ces caractéristiques sont ensuite analysées par un réseau de neurones classifieur, qui donne une classe à chaque pixel.

Les auteurs de [Wang et al., 2016] proposent un procédé d'apprentissage multitâche. Un premier réseau de neurones construit des cartes de chaleur décrivant la probabilité de la position des articulations, qui sont ensuite transmises à un réseau de neurones qui effectue l'inférence de la posture 3D en cherchant la configuration optimale des partie du corps, en utilisant à la fois l'apparence et la compatibilité géométrique. De même, la méthode proposée dans [Haque et al., 2016] détecte d'abord les parties locales du corps, puis une posture globale est produite de manière itérative en utilisant un réseau de neurones convolutif récurrent et contenant un module de mémoire à long et à court terme (en anglais *long short term memory*, abrégé LSTM).

Une approche présentée par [Moon et al., 2018] transforme l'image de profondeur en un

ensemble de voxels et utilise un réseau de neurones voxel vers voxel pour l'estimation de la posture des mains et du corps. Ainsi, l'espace 3D est divisé en une grille de voxel, et des cartes de chaleur volumétriques définissent la probabilité de chaque voxel de contenir les articulations associées.

Enfin, les auteurs de [Marín-Jiménez et al., 2018] présentent un modèle qui utilise un réseau de neurones convolutif pour calculer des poids permettant d'estimer la posture 3D comme une combinaison linéaire de postures prototypes.

Si ces approches ont fait leurs preuves sur des ensembles de données publics, ou dans des cas d'utilisation classiques pour l'estimation de posture sur les images de profondeur, elles ne semblent cependant pas adaptées à notre cas particulier d'utilisation en milieu industriel, avec des images de profondeur bruitées et dégradées. En effet, les modèles proposés sont souvent très profonds, avec un très grand nombre de paramètres, ce qui les rend coûteux à entraîner, ou bien ils sont utilisés sur des images très différentes des notres, qui ne contiennent en particulier pas les dégradations impliquées par l'environnement industriel. Dans notre situation, nous préférons construire des modèles légers et rapides à entraîner. C'est pour cette raison que nous avons décidé de mener l'étude présentée dans la partie suivante.

5.3 Influence de l'architecture du réseau de neurones sur l'estimation de posture

Cette partie présente une analyse des résultats de l'apprentissage d'un ensemble de réseaux de neurones afin de déterminer l'influence de différents hyper-paramètres pour l'estimation de la posture.

Au cours de cette étude, afin de simplifier les calculs et pour des raisons de redondance, la position du nez étant en général très proche de la position de la tête, tout comme celles du pouce et de l'extrémité de la main sont proches de la position de la main, nous restreignons l'estimation de posture 3D à l'estimation de la position 3D des quinze articulations suivantes (voir Figure 5.14) :

- Bas de la colonne ;
- Milieu de la colonne ;
- Haut de la colonne ;
- Bassin gauche ;
- Bassin droit ;
- Cou ;
- Tête ;
- Épaule gauche ;
- Épaule droite ;
- Coude gauche ;
- Coude droit ;
- Poignet gauche ;

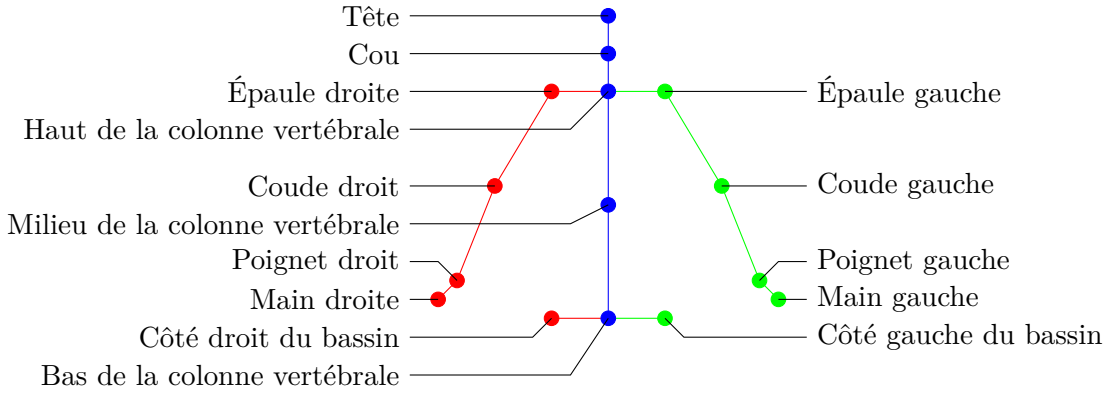


FIGURE 5.14 – Les quinze articulations utilisées pour définir la posture 3D du haut du corps. Dans cette étude, estimer la posture 3D revient à estimer la position 3D de ces quinze points. Par rapport aux annotations complètes, on a supprimé les pouces, les bout des mains et le nez.

— Poignet droite ;
— Main gauche ;
— Main droite.

Pour cette étude, on utilise les images de profondeur de synthèse telles que celles présentées en Section 5.1 et Figures 5.6, 5.7 et 5.8.

Les réseaux prennent en entrée une image de taille 64×64 (voir Figure 5.15). Pour des questions pratiques déjà exposées précédemment, c'est-à-dire dans le but d'optimiser le coût de mise en place du dispositif dans des conditions réelles, nous avons choisi de diminuer la résolution des images d'entrée. En sortie, on récupère la position 3D (x, y, z) de chacune des articulations, soit un vecteur de dimension quarante-cinq (3×15). On utilise ici les *coordonnées réelles*, définies en Section 5.1.

Les mesures affichées sont calculées sur un ensemble de test, distinct des données utilisées pour l'entraînement des réseaux de neurones. Afin d'estimer la distance à la vérité dans chacune des directions de l'espace cartésien, nous mesurons l'erreur absolue moyenne entre la posture réelle y et la posture estimée par le réseau de neurones $\widehat{y}$:

$$EAM(y, \widehat{y}) = \frac{1}{n} \sum_{i=1}^{n} |y_i - \widehat{y}_i|, \text{ avec : } \begin{cases} |y_i - y_j| = \frac{1}{m} \sum_{k=1}^{m} |y_{i,k} - y_{j,k}| \\ n \quad \text{le nombre de données de test} \\ m \quad \text{le nombre d'articulations à estimer} \\ y_{i,k} \text{ et } y_{j,k} \in \mathbb{R}^3 \end{cases} \tag{5.1}$$

L'erreur absolue pour une articulation a d'une estimation de posture est donc la suivante :

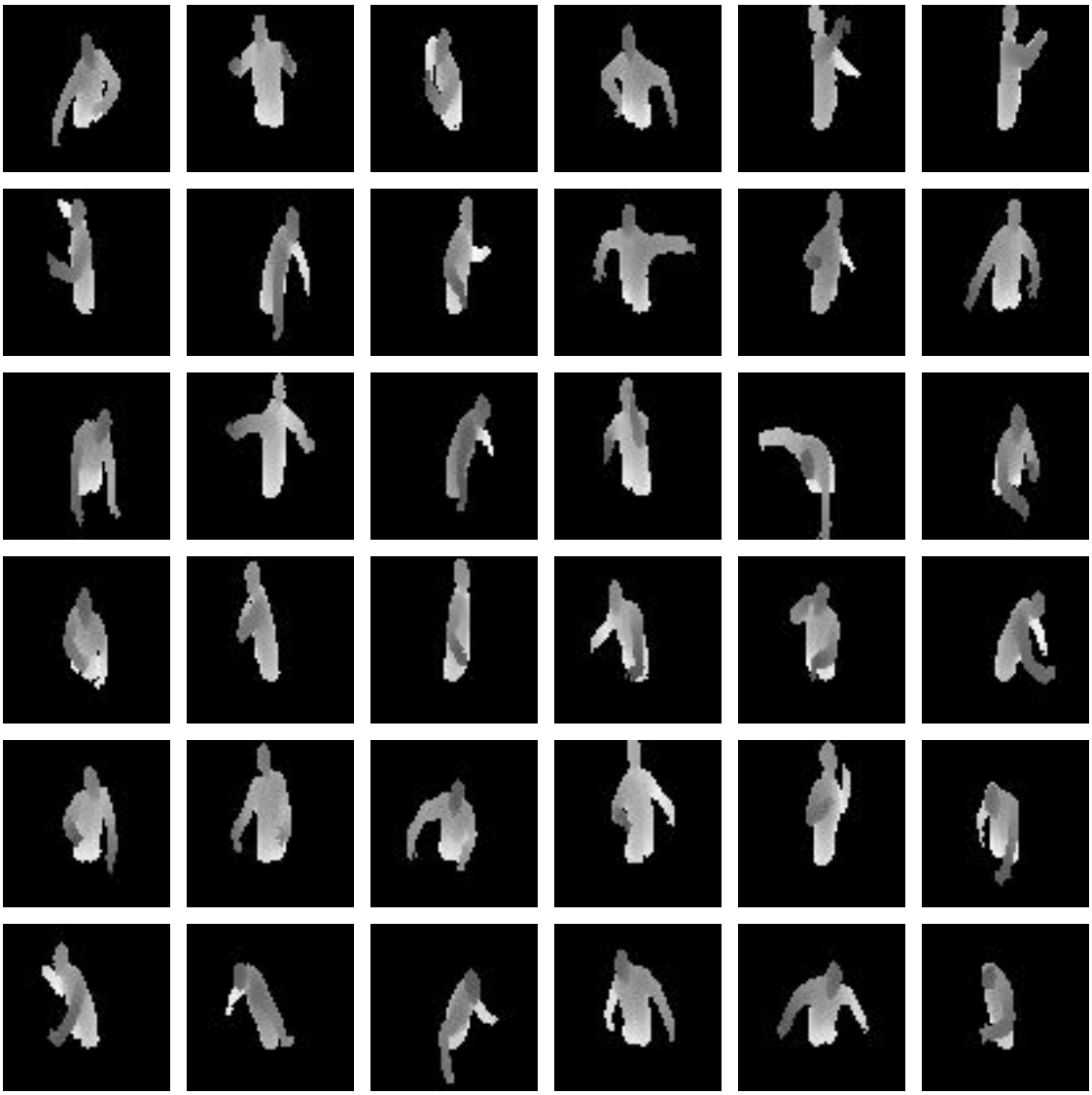


FIGURE 5.15 – Exemple d'images de synthèse dont on souhaite estimer la posture. Notre ensemble de données contient 200 000 images annotées. Ces images ont été redimensionnées par rapport à la Figure 5.6.

$$EA(a, \widehat{a}) = \frac{1}{3} \sum_{k=1}^{3} |a_k - \widehat{a}_k|$$

Ainsi, pour une erreur d'un centimètre (dix millimètres) de chaque articulation dans chacune des coordonnées, l'erreur absolue pour une articulation serait égale à :

$$EA(a, \widehat{a}) = \frac{1}{3} \sum_{k=1}^{3} 10 = 10 \text{ mm}$$

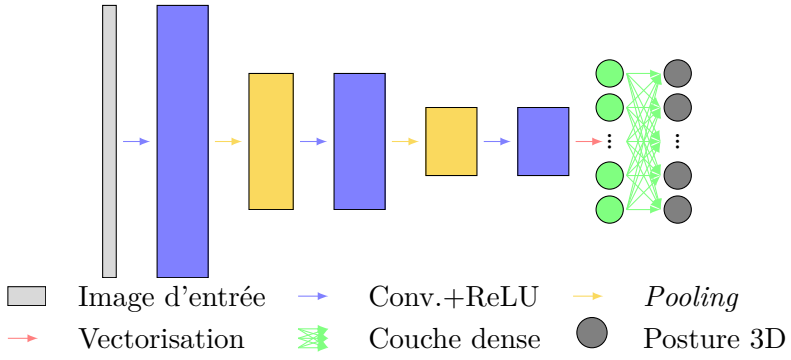


FIGURE 5.16 – Exemple d'architecture de réseau de neurones d'estimation de posture 3D. Ce réseau est constitué de trois couches covolutives utilisant une activation ReLU (flèches bleues), séparées par des couches de *pooling* (flèches jaunes). Une couche dense finale (flèches vertes) effectue la régression des caractéristiques extraites de l'image vers une posture 3D composée de 45 valeurs représentant les quinze articulations qui définissent la posture. Entre la dernière couche convolutive et la couche dense, on réalise une opération qui permet de vectoriser le tenseur des caractéristiques obtenu par la dernière couche convolutive (flèche rouge).

L'écart global serait identique du fait du moyennage. Ainsi, on peut estimer que l'erreur pour chaque articulation pour chacune des coordonnées (x, y, z) est environ égale à la valeur de l'erreur absolue moyenne. Il est cependant important de noter que dans la réalité, certaines articulations ont une erreur très inférieure à la moyenne, tandis que d'autres ont une erreur très supérieure. Ces articulations avec une erreur importante sont en général des articulations avec une plus grande variabilité dans les positions possibles (par exemple les poignets) que ceux avec une faible erreur (telles que les articulations de la colonne vertébrale).

Pour évaluer l'influence des paramètres, nous comparons les pertes et les mesures d'erreur absolue moyenne des réseaux de neurones. Nous utilisons 90% des données (soit 180 000 images) comme données d'entraînement, et réservons les 10% de données restantes (soit 20 000 images) pour les tests. Les courbes affichées dans la suite de cette étude ont été construites en mesurant l'erreur absolue moyenne sur les données de test.

Les réseaux de neurones que nous utilisons ont tous la même structure générale. Ce sont des réseaux de neurones convolutifs, où les couches de convolution sont séparées par des couches d'activations de type *Rectified Linear Unit* (ou ReLU), de potentielles couches de *pooling* et l'estimation finale est effectuée grâce à une couche dense. Un exemple d'un tel modèle est présenté en Figure 5.16.

Les erreurs absolues moyennes pour chacune des articulations à estimer sont données en Figure 5.17. On observe que la position des articulations du tronc semble plus facile à estimer que celle des membres supérieurs (comme les bras et les mains). On voit donc que les articulations avec la plus grande variabilité de position (voir Section 5.1.3) sont les

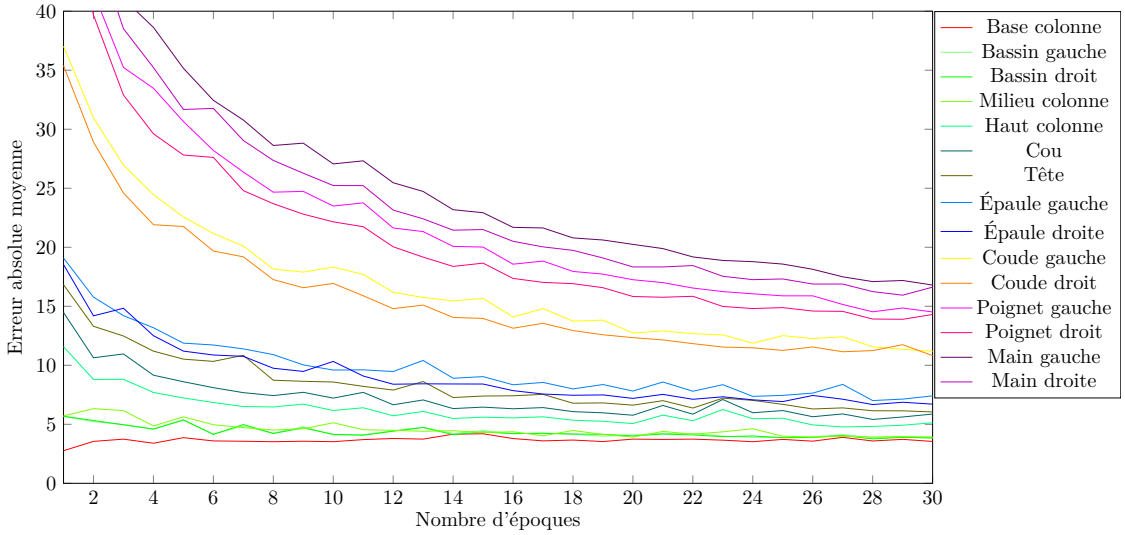


FIGURE 5.17 – Évolution de l'erreur absolue moyenne par articulation sur un des réseaux de l'étude pendant l'apprentissage. On peut distinguer plusieurs groupements : les articulations avec une distance très faible (articulations de la colonne vertébrale, bassin, épaules, cou et tête), et des articulations dont la distance augmente à mesure qu'on s'éloigne du tronc : coudes (en jaune/orange), poignets (en rose) et mains (en violet).

articulations donc la position est la plus difficile à estimer : il y a donc une corrélation entre la difficulté d'estimation d'une articulation et sa variabilité. On remarque par ailleurs une symétrie droite/gauche.

Ces observations sont valables pour tous les réseaux que nous avons entraînés, ainsi nous n'utilisons que la perte globale pour comparer les performances des différents réseaux.

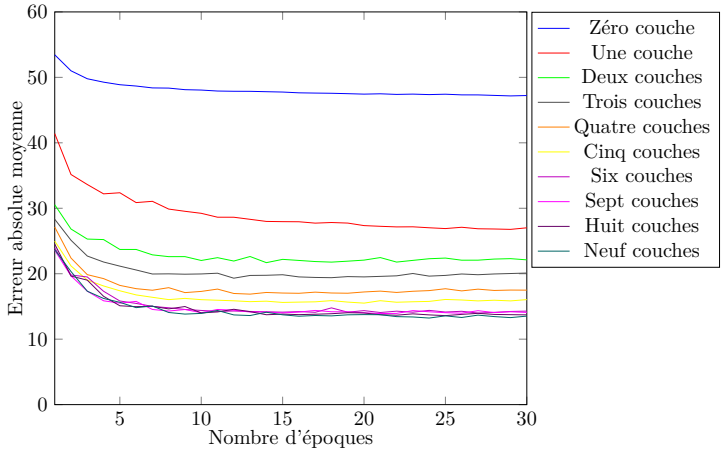
Les études sur l'influence des différents paramètres sont décrites dans les paragraphes suivants.

Sauf mention contraire, nous utilisons un réseau de neurones convolutif contenant trois couches de convolution de soixante-quatre filtres de taille 3×3, dont l'entraînement est effectué avec un optimiseur Adam [Kingma and Ba, 2015] avec ses paramètres par défaut, et qui utilise la distance euclidienne moyenne comme fonction de perte. Les valeurs de profondeur des données fournies en entrée du modèle sont par défaut normalisées dans l'intervalle [-1 ;1] à l'aide d'un *Min-Max Scaling*, qui est l'intervalle recommandé pour faire fonctionner au mieux les filtres des couches de convolution.

5.3.1 Nombre de couches du réseau

Le premier paramètre que nous étudions est le nombre de couches convolutives composant le réseau, c'est-à-dire sa profondeur.

Ainsi, on fait varier le nombre de couches d'un lot de réseaux, tous les autres paramètres



Nombre de couches	EAM
Aucune couche	47.2
Une couche	27.0
Deux couches	22.1
Trois couches	20.1
Quatre couches	17.5
Cinq couches	16.0
Six couches	14.3
Sept couches	14.1
Huit couches	13.7
Neuf couches	**13.5**

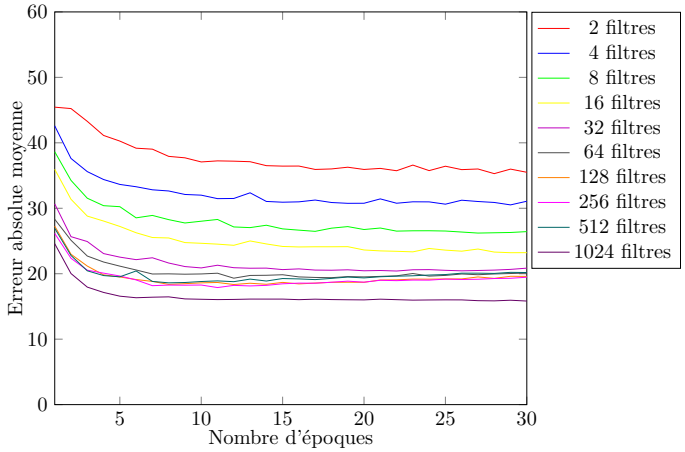
FIGURE 5.18 – Évolution de l'erreur absolue moyenne de différents réseaux de neurones dont on fait varier le nombre de couches de convolution. Des valeurs numériques en fin d'apprentissage pour chacune des deux mesures sont données dans les tableaux de la colonne de droite. On peut voir que le nombre de couches a une influence très importante sur la qualité de l'apprentissage, mais que l'influence de l'ajout d'une couche diminue avec le nombre de couches et que l'amélioration des performances n'est plus significative une fois qu'on dépasse six couches convolutives.

(taille et nombre de filtres, par exemple) étant fixés et égaux par ailleurs. Les résultats sont donnés en Figure 5.18.

On observe que le nombre de couches à l'intérieur du réseau a une influence très grande sur les performances de celui-ci. Cette influence diminue cependant au fur et à mesure que le nombre de couches augmente. En effet, s'il est possible de distinguer les réseaux avec peu de couches (dont le perceptron mono-couche avec les plus mauvais résultats), on a plus de mal à différencier les pertes des réseaux contenant un plus grand nombre de couches.

Cette influence de la profondeur s'explique par le type d'information extraites par les couches successives des réseaux de neurones : si les premières couches ne sont qu'en mesure d'extraire des caractéristiques simples (telles que des contours) à petite échelle, les couches suivantes combinent les caractéristiques extraites par les couches précédentes et construisent des concepts de plus en plus abstraits (par exemple des motifs, puis des parties du corps, etc.) à des échelles plus importantes.

Il semble cependant que le problème que nous cherchons à résoudre ne demande pas un (trop) fort niveau d'abstraction, car les réseaux qui utilisent plus de six couches présentent des performances équivalentes.



Nombre de filtres	EAM
2 filtres	35.5
4 filtres	31.1
8 filtres	26.4
16 filtres	23.2
32 filtres	20.9
64 filtres	20.1
128 filtres	19.6
256 filtres	19.5
512 filtres	20.2
1024 filtres	**15.8**

FIGURE 5.19 – Évolution de l'erreur absolue moyenne (distance L1) pour différents réseaux de neurones où l'on fait varier le nombre de filtres des couches de convolution. On constate que le nombre de filtres a une influence très importante, mais que celle-ci diminue avec l'augmentation du nombre de filtres et l'amélioration des performances n'est plus significative. Par ailleurs, on observe un surapprentissage pour certains réseaux utilisant un grand nombre de filtres (en rose et vert foncé).

5.3.2 Nombre de filtres par couche

Dans cette section, nous présentons les résultats à propos de l'influence du nombre de filtres sur la qualité de l'estimation de la posture. Pour cela, nous avons entraîné un lot de réseaux de neurones dont le seul paramètre qui varie est le nombre de filtres dans les couches de covolution. Dans un premier temps, le nombre de filtres est constant pour chaque couche du réseau. Dans une seconde expérience, on fait varier le nombre de filtres pour les couches d'un même réseau.

Réseaux à nombre de filtres constant

Dans cette section, on s'intéresse à l'influence du nombre de filtres dans un réseau de neurones. Les couches ont dans cette partie le même nombre de filtres.

Les résultats pour l'erreur absolue moyenne globale et la distance L1 sont donnés en Figure 5.19. On observe que la perte diminue avec l'augmentation du nombre de filtres. Cependant, cette influence diminue au fur et à mesure que cette quantité augmente, et les réseaux utilisant le plus grand nombre de filtres présentent des signes de surapprentissage (voir un exemple illustratif en Figure 5.20).

Suite à cette expérience, le réseau qui semble le plus intéressant, c'est-à-dire avec les meilleurs résultats mais sans surapprentissage, est le réseau qui utilise des couches à 1024 filtres.

L'influence du nombre de filtres s'explique par le nombre de caractéristiques calculables :

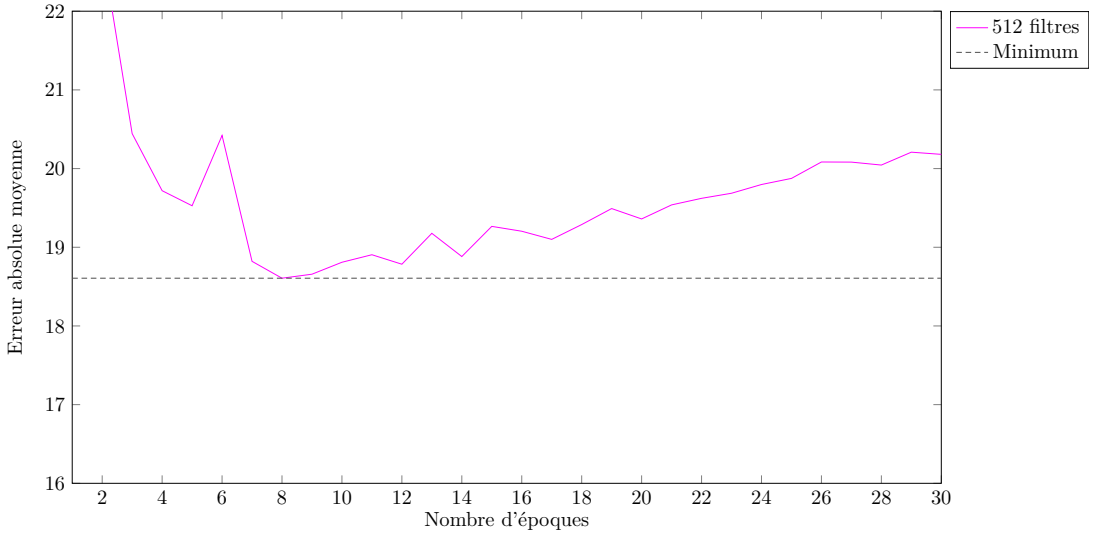


FIGURE 5.20 – Zoom sur l'évolution de l'erreur quadratique sur le réseau utilisant 512 filtres par couche. Le minimum est donné par la courbe en gris foncé. À partir des epochs 8 à 15, on observe du surapprentissage, repérable par une augmentation de la perte sur les données de test.

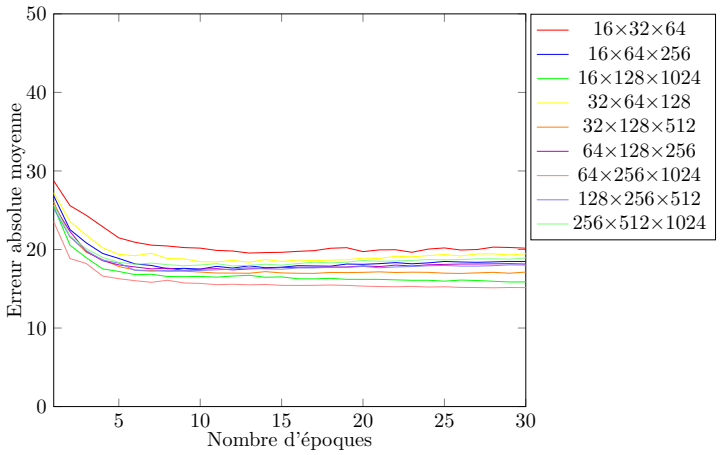
plus le nombre de filtres par couche est important, plus le nombre de caractéristiques extraites de l'image (pour la première couche) ou des caractéristiques précédentes est important et donc plus le réseau peut extraire d'information. En analysant plus d'information, le réseau de neurones peut construire des concepts qui correspondent mieux au problème à résoudre et donc être plus performant. Ceci explique aussi le surapprentissage qui apparaît parfois : le réseau est si puissant qu'il est en mesure de s'adapter aux données d'apprentissage plus qu'il n'en est nécessaire, et cela devient néfaste à sa capacité de généralisation.

Réseaux à nombre de filtres variable

Dans cette section, on s'intéresse de nouveau à l'influence du nombre de filtres dans le réseau de neurones. Cette fois-ci, les couches ont un nombre de filtres variable.

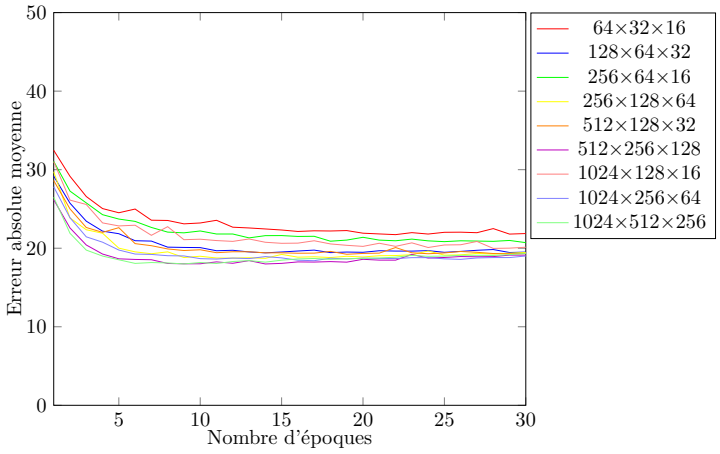
Les résultats pour la distance L1 sont donnés en Figures 5.21 pour un nombre de filtres croissant et Figures 5.22 pour un nombre de filtres décroissant. On observe, dans le cas des réseaux avec un nombre de filtres décroissant, une très légère amélioration. Cependant, pour les réseaux utilisant un nombre de filtres croissant et ceux qui utilisent un grand nombre de filtres, on distingue des symptômes de surapprentissage.

D'après les tests que nous venons de présenter, il semble que faire varier le nombre de filtres dans les couches de notre réseau de neurones ne semble pas améliorer significativement ses performances. Ceci apparaît malgré tout contre-intuitif. On pouvait rai-



Nombre de filtres	EAM
16×32×64	20.2
16×64×256	18.0
16×128×1024	15.9
32×64×128	19.5
32×128×512	17.1
64×128×256	18.1
64×256×1024	**15.1**
128×256×512	18.0
256×512×1024	18.9

FIGURE 5.21 – Évolution de l'erreur absolue moyenne de différents réseaux de neurones où l'on fait varier le nombre de filtres de chaque couche de convolution. On constate que le nombre de filtres a une influence qui semble très légère. De plus, la plupart des réseaux présentent des signes de surapprentissage.



Nombre de filtres	EAM
64×32×16	21.9
128×64×32	19.5
256×64×16	20.7
256×128×64	19.5
512×128×32	19.3
512×256×128	**19.1**
1024×128×16	20.1
1024×256×64	**19.0**
1024×512×256	**19.2**

FIGURE 5.22 – Évolution de la distance L1 de différents réseaux de neurones où l'on fait varier le nombre de filtres de chaque couche de convolution. On constate que le nombre de filtres a une influence, mais elle semble relativement faible. On peut remarquer, dans certains cas, une légère détérioration, alors que d'autres montrent une petite amélioration. On observe cependant le même surapprentissage que sur l'expérience précédente pour les réseaux avec un grand nombre de filtres.

Nombre de filtres de la dernière couche	EAM nombre des filtres décroissant	EAM nombre des filtres constant	EAM nombre des filtres croissant
16 filtres	**20.7**	23.2	
32 filtres	**19.3**	20.9	
64 filtres	**19.0**	20.1	20.2
128 filtres	**19.1**	19.6	19.5
256 filtres	19.2	19.5	**18.0**
512 filtres		20.2	**17.1**
1024 filtres		15.8	**15.1**

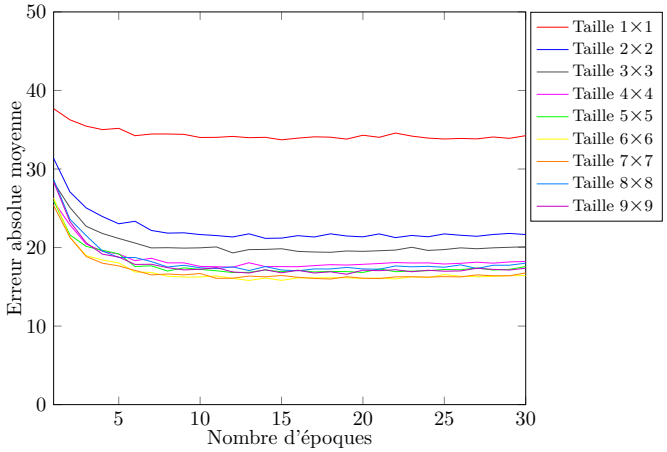
Nombre de filtres de la première couche	EAM nombre des filtres décroissant	EAM nombre des filtres constant	EAM nombre des filtres croissant
16 filtres		23.2	**15.9**
32 filtres		20.9	**17.1**
64 filtres	21.9	20.1	**15.1**
128 filtres	19.5	19.6	**18.0**
256 filtres	19.5	19.5	**18.9**
512 filtres	**19.1**	20.2	
1024 filtres	19.0	**15.8**	

TABLE 5.3 – Erreur absolue moyenne finale pour différents réseaux de neurones où l'on fait varier le nombre de filtres des couches de convolution. On a sélectionné les meilleures performances en fin d'entraînement pour chaque nombre de filtres et chaque type d'évolution. On constate que le nombre de filtres a une influence très importante, mais que celle-ci diminue avec l'augmentation du nombre de filtres. La meilleure performance pour chaque nombre de filtres est affichée en gras. Il est difficile de déterminer le type d'évolution (nombre de filtres décroissant, constant ou croissant) le plus efficace, mais les réseaux avec une couche finale identique (même nombre de filtres) semblent se comporter de la même manière.

sonnablement s'attendre à une amélioration des performances avec un nombre de filtres variable : selon le problème, il peut sembler plus intéressant d'extraire de nombreuses caractéristiques de bas niveau (à l'aide des premières couches) ou de haut niveau (à l'aide des couches les plus profondes). Dans notre cas, un nombre de filtres constant obtient des résultats semblables aux réseaux qui contiennent le même nombre de filtres en couche finale (voir Tableau 5.3).

5.3.3 Influence de la taille des filtres

Dans cette section, les couches de convolution ont le même nombre de filtres, mais la taille de ceux-ci varie. Comme précédemment, cette étude se décompose en deux expé-



Taille des filtres	EAM
1×1	34.2
2×2	21.7
3×3	20.1
4×4	18.2
5×5	17.6
6×6	**16.4**
7×7	16.8
8×8	18.0
9×9	17.4

FIGURE 5.23 – Évolution de l'erreur absolue moyenne de différents réseaux de neurones où l'on fait varier la taille des filtres de convolution. On observe que les réseaux utilisant des filtres de grande taille sont sujets à du surapprentissage (violet, bleu clair, jaune, orange et vert). En revanche, les filtres de trop petite taille (en rouge) ne permettent pas un apprentissage correct. Les filtres de tailles moyenne (gris, bleu foncé et rose) semblent amener à des performances très proches, avec un léger avantage pour le réseaux utilisant des filtres de taille 4×4 (en rose).

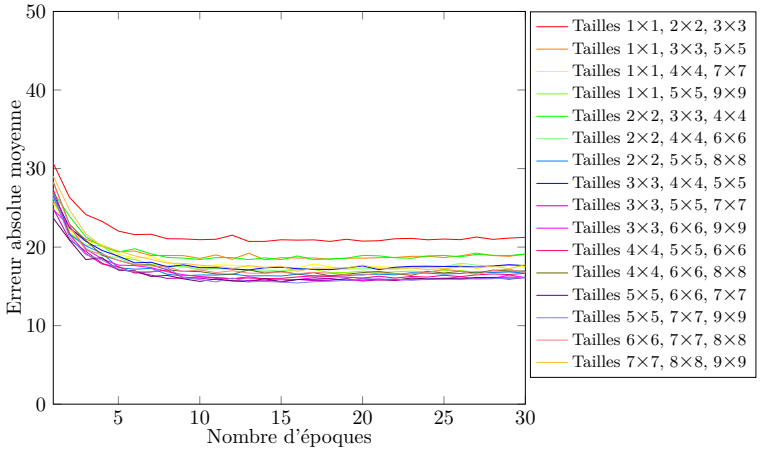
riences : la taille des filtres est tout d'abord constante au sein d'un même réseau, puis elle varie entre les couches.

Réseaux à taille de filtre constante

Dans cette partie, on étudie l'influence de la taille des filtres dans les couches convolutives. Ici, tous les filtres de toutes les couches ont une taille identique.

Les résultats pour l'erreur absolue moyenne sont donnés en Figure 5.23.

On observe que la taille des filtres a une grande influence sur la qualité de l'estimation de posture par le réseau de neurones. En effet, des réseaux utilisant des filtres de petite taille ont une performance nettement inférieure à ceux utilisant des filtres de taille plus importante. Comme lors des expériences précédentes, on observe que l'influence de la taille du filtre diminue avec son augmentation : les filtres de taille supérieure à 5×5 ont tous des performances équivalentes, avec une efficacité maximale pour les filtres de taille 6×6 et qui décroit pour des tailles supérieures. Cela s'explique par la petite taille de nos images d'entrée (64×64) et de la répartition de l'information contenue dans celles-ci : l'utilisateur dont on souhaite estimer la posture est situé au centre de l'image et n'occupe pas toute l'image (pour rappel, voir Figure 5.15).



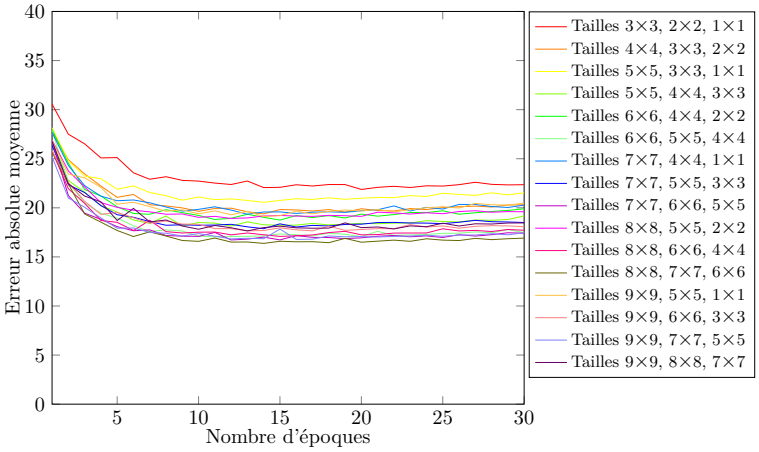
Tailles des filtres	EAM
1×1,2×2,3×3	21.2
1×1,3×3,5×5	19.1
1×1,4×4,7×7	17.5
1×1,5×5,9×9	16.9
2×2,3×3,4×4	19.1
2×2,4×4,6×6	17.6
2×2,5×5,8×8	16.7
3×3,4×4,5×5	17.6
3×3,5×5,7×7	**16.2**
3×3,6×6,9×9	**16.1**
4×4,5×5,6×6	16.7
4×4,6×6,8×8	16.9
5×5,6×6,7×7	**16.1**
5×5,7×7,9×9	**16.1**
6×6,7×7,8×8	16.7
7×7,8×8,9×9	17.8

FIGURE 5.24 – Évolution de l'erreur absolue moyenne de réseaux de neurones où l'on fait augmenter la taille des filtres de convolution à chaque couche. On observe que les réseaux utilisant des filtres de petite taille au départ et une croissance forte ont des performances légérement meilleures que lorsqu'on utilise des filtres de grande taille dès le départ. La meilleure efficacité semble atteinte par des réseaux qui commencent avec une taille de filtre moyenne et finissent avec une taille de filtre de 9×9.

Réseaux à taille de filtre variable

Dans cette partie, les couches de convolution d'un même réseau ont une taille de filtres variable. Les résultats de mesure de l'erreur absolue moyenne sont donnés en Figures 5.24 et 5.25. On voit que les performances des réseaux sont relativement proches (les écarts sont tous compris entre 22.4 et 16.1). On peut tout de même remarquer que, pour les filtres de taille croissante, les meilleurs résultats sont généralement obtenus avec les filtres de taille 9×9 ou 8×8 sur la couche finale et une croissance importante (2×2 ou 3×3). En revanche, pour les filtres de taille décroissante, les meilleurs résultats sont obtenus pour des filtres de grande taille sur la première couche (8×8 ou 7×7) avec une décroissance lente (1×1).

Suite aux résultats des deux études précédentes, on compare les performances en fonction de la taille des filtres de la première couche et de la dernière couche (voir Tableau 5.4 et 5.26). On observe que les réseaux qui présentent les écarts les plus faibles par rapport à la vérité terrain sont les réseaux dont la taille moyenne des filtres est égale à 5×5 ou 6×6 (voire 7×7). Ceci confirme notre étude sur les filtres de taille constante, où le réseau le



Tailles des filtres	EAM
3×3,2×2,1×1	22.4
4×4,3×3,2×2	20.3
5×5,3×3,1×1	21.5
5×5,4×4,3×3	19.2
6×6,4×4,2×2	19.9
6×6,5×5,4×4	**17.5**
7×7,4×4,1×1	20.3
7×7,5×5,3×3	18.5
7×7,6×6,5×5	**17.4**
8×8,5×5,2×2	19.5
8×8,6×6,4×4	17.7
8×8,7×7,6×6	16.9
9×9,5×5,1×1	20.5
9×9,6×6,3×3	18.1
9×9,7×7,5×5	**17.5**
9×9,8×8,7×7	18.5

FIGURE 5.25 – Évolution de l'erreur absolue moyenne de différents réseaux de neurones où l'on fait diminuer la taille des filtres de convolution à chaque couche. La meilleure efficacité semble atteinte par le réseau qui commence avec une taille de filtre de 8×8 et qui termine avec une taille de filtre de 6×6 et une taille qui décroit de 1×1 par couche.

plus performant était le réseau aux filtres de taille 6×6. On note par ailleurs que très peu de réseaux font mieux que celui-ci : il n'y en a aucun avec une taille de filtre décroissante, et quatre avec une taille de filtre croissante, dont les progrès sont très faibles : on passe d'une erreur de 16.4 à une erreur de 16.1, soit une amélioration de moins de 2%. Ainsi, il ne semble pas très intéressant de faire varier la taille des filtres dans un réseau de neurones cherchant à résoudre notre problème.

5.3.4 Réduction de la dimension spatiale des caractéristiques

Dans cette section, on s'intéresse à l'influence de la présence de couches de *pooling* ("mise en commun") dans le réseau de neurones, ainsi que celle de l'utilisation de *stride* dans les couches convolutives.

Les couches de *pooling* et le *stride* dans les couches convolutives ont tous les deux pour but de réduire la dimension spatiale des caractéristiques construites par les différentes couches convolutives du réseau de neurones. Les couches de *pooling* sont une forme de sous-échantillonnage des caractéristiques. Chaque caractéristique est découpée en tuile, et le signal en sortie de la tuile est défini en fonction du type de couche utilisé. On trouve généralement les couches de *max-pooling*, où la valeur de sortie de la tuile est la valeur maximale parmi les valeurs de la tuile, et les couches de *average-pooling*, où la valeur de

Taille des filtres de la dernière couche	EAM taille des filtres décroissante	EAM taille des filtres constante	EAM taille des filtres croissante	Taille moyenne des filtres du meilleur réseau
1×1	**20.3**	34.2		4×4
2×2	**19.5**	21.7		5×5
3×3	**18.1**	20.1	21.4	6×6
4×4	**17.5**	18.2	19.1	5×5
5×5	17.4	**17.2**	17.6	5×5
6×6	16.9	**16.4**	16.7	6×6
7×7	18.5	16.8	**16.1**	6×6
8×8		18.0	**16.7**	7×7
9×9		17.4	**16.1**	7×7

Taille des filtres de la dernière couche	EAM taille des filtres décroissante	EAM taille des filtres constante	EAM taille des filtres croissante	Taille moyenne des filtres du meilleur réseau
1×1		34.2	**16.9**	5×5
2×2		21.7	**16.7**	5×5
3×3	22.4	20.1	**16.1**	6×6
4×4	20.3	18.2	**16.7**	5×5
5×5	19.2	17.2	**16.1**	6×6 et 7×7
6×6	17.5	**16.4**	16.7	6×6
7×7	17.4	**16.8**	17.8	7×7
8×8	**16.9**	18.0		6×6
9×9	17.5	**17.4**		9×9

TABLE 5.4 – Erreur absolue moyenne finale pour différents réseaux de neurones où l'on fait varier la taille des filtres des couches de convolution. On affiche uniquement les meilleures performances en fin d'entraînement pour chaque type d'évolution et chaque taille de filtre. On constate que la taille des filtres a une influence non négligeable. La meilleure performance pour chaque taille de filtre est affichée en gras. Il semble que les réseaux les plus performants sont les réseaux dont la taille moyenne est 5×5 ou 6×6 (voire 7×7).

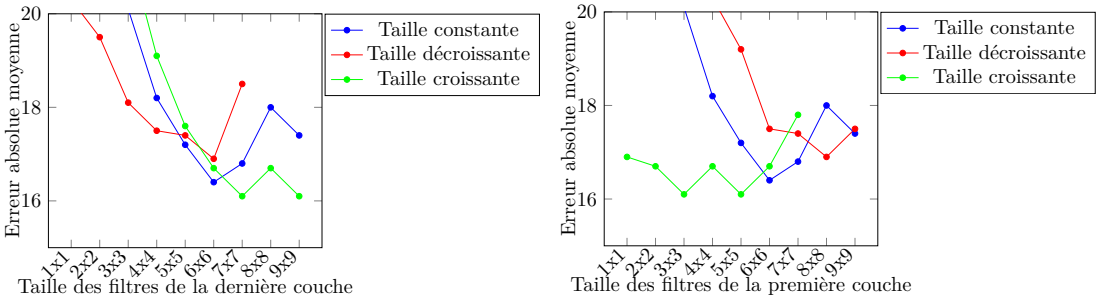


FIGURE 5.26 – Évolution de l'écart mesuré par la distance L1 en fonction de la taille des filtres de la dernière couche (à gauche) et de la première couche (à droite). On observe, dans les deux cas, une inflexion au niveau des filtres de taille 6×6 et un changement du type d'évolution de la taille de filtres la plus efficace entre les tailles 5×5 et 7×7.

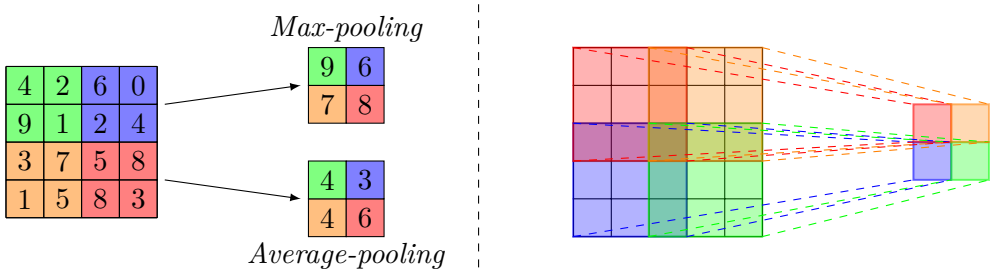
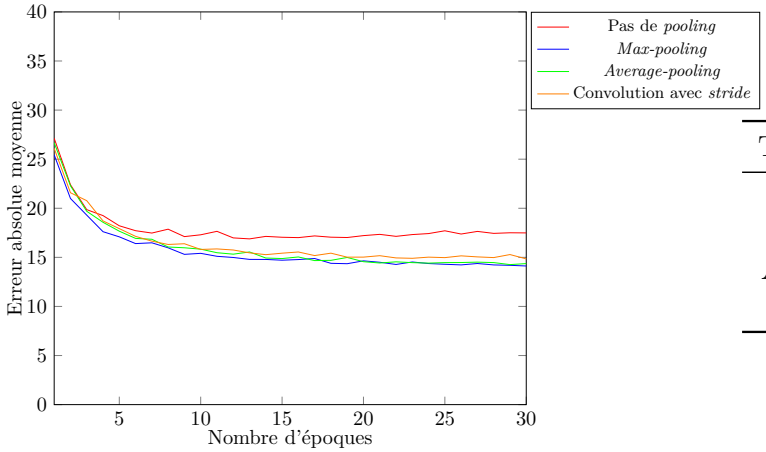


FIGURE 5.27 – Exemple de *pooling* (à gauche) et de *stride* (à droite) dans un réseau de neurones convolutif. Le *pooling* permet de réduire la dimension spatiale des caractéristiques en sélectionnant le maximum (*max-pooling*) ou la moyenne (*average-pooling*) d'un patch glissant sur l'image, alors que le *stride* d'une couche de convolution définit le pas de déplacement des filtres de celle-ci. Ces deux types d'opération permettent de réduire la dimension spatiale des caractéristiques.

sortie de la tuile est égale à la moyenne des valeurs de la tuile. Le *stride* d'une couche de convolution est le pas de déplacement du filtre de convolution sur la caractéristique, défini verticalement et horizontalement. Ces deux outils de réduction de la dimension spatiale des caractéristiques sont présentés en Figure 5.27.

Dans cette partie, les réseaux de neurones sont nommés par rapport au nombre de filtres de convolution et aux opérations (*pooling* ou *stride*) utilisées. Par exemple, un réseau de neurones nommé 64^2 MP 64 est un modèle dont l'architecture est la suivante :

$$Conv(64) -> Conv(64) -> MaxPooling -> Conv(64) -> Dense$$



Type de *pooling*	EAM
Pas de *pooling*	17.3
Max-pooling	**14.1**
Average-pooling	14.4
Stride	14.8

FIGURE 5.28 – Évolution de la perte lorsqu'on ajoute différents types de couche de *pooling* ou une couche de convolution avec *stride*. On observe que la présence d'une couche de max-pooling et l'utilisation de *stride* améliorent la qualité des résultats. En effet, l'ajout d'une couche de *pooling* au milieu du réseau à quatre couches permet de faire diminuer l'erreur absolue moyenne (vert et bleu). De même, utiliser du *stride* sur la deuxième couche de convolution du réseau à quatre couches permet d'améliorer les résultats. Après cette expérience, il est cependant difficile de distinguer quel type de *pooling* est le plus adapté à notre problème, même si le *pooling* semble avoir un léger avantage sur les convolutions avec *stride*.

La Figure 5.28 présente les performances de différents réseaux de neurones, en fonction de la présence ou de l'absence de couches de *pooling* ou de *stride* dans les couches de convolution.

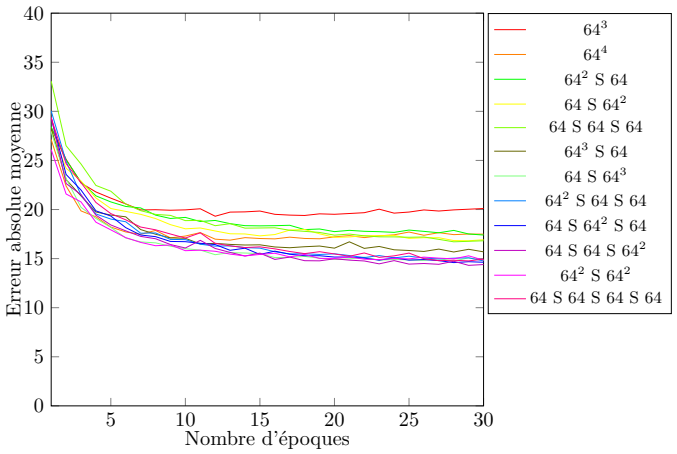
On observe qu'ajouter des opérations qui permettent de réduire la dimension spatiale des caractéristiques améliore les résultats obtenus par le réseau de neurones.

Les sections suivantes s'intéressent à l'influence de chacun de ces outils. Tous les modèles présentés dans cette partie utilisent un noyau de taille 2×2 pour le *pooling* et un pas $(2, 2)$.

Ajout de *stride* dans les couches convolutives

Dans cette partie, on étudie l'influence de l'ajout de *stride* dans les couches de convolution. Les couches qui utilisent du *stride* ont un pas $(2, 2)$, ce qui permet de diviser par deux la hauteur et la longueur des caractéristiques. Les résultats sont donnés en Figure 5.29.

On observe que l'utilisation de *stride* permet d'améliorer les performances finales du réseau de neurones. Cependant, l'erreur absolue moyenne ne diminue pas de manière significative lorsqu'on augmente le nombre de couches de convolution avec *stride*. Ainsi, il semble qu'une unique couche de convolution avec *stride* soit suffisante. On remarque de plus qu'il semble plus intéressant de positionner cette couche en début de réseau. Le *stride*, en réduisant la taille des caractéristiques extraites de l'image, permet de réduire le risque



Réseau	EAM
64^3	20.1
64^4	17.5
64^2 S 64	17.4
64 S 64^2	16.9
64 S 64 S 64	16.8
64^3 S 64	15.7
64^2 S 64^2	14.8
64 S 64^3	14.8
64^2 S 64 S 64	14.8
64 S 64^2 S 64	14.6
64 S 64 S 64^2	**14.4**
64 S 64 S 64 S 64	14.9

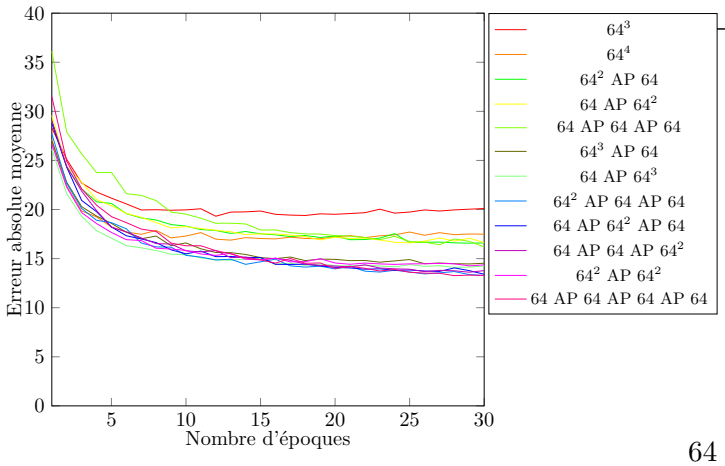
FIGURE 5.29 – Évolution de l'erreur absolue moyenne lorsqu'on ajoute des couches de convolution avec *stride*. Le réseau est défini par le nombre de filtres des couches convolutives et leur nombre en exposant. S signifie que la couche précédente utilise un *stride* de taille $(2, 2)$. On observe que la présence de couches de convolution avec *stride* améliore grandement la qualité des résultats. En effet, si on ajoute des couches de convolution avec *stride* au réseau à trois couches, on arrive à atteindre des performances supérieures à celles du réseau à quatre couches. De même, l'ajout de couches de convolution avec *stride* au réseau à quatre couches améliore là aussi ses résultats. D'après cette expérience, il semble qu'une unique couche de convolution avec *stride* soit suffisante pour améliorer les performances.

de surapprentissage, ce qui permet au modèle d'augmenter sa capacité de généralisation, et améliore donc les résultats obtenus.

Ajout de couches d'*average-pooling*

Dans cette partie, on étudie l'impact de l'ajout de couches de *average-pooling*. Les couches de *average-pooling* utilisées ont un noyau de taille 2×2 et un pas $(2, 2)$, c'est-à-dire qu'elles sélectionnent la valeur moyenne de patchs de caratéristique de taille 2×2 et divisent pas deux la hauteur et la longueur des caractéristiques. Les résultats sont donnés en Figure 5.30.

On observe que l'utilisation de couches d'*average-pooling* permet d'améliorer les performances du modèle. De plus, il semble qu'utiliser plusieurs couches de *average-pooling* permet d'obtenir de meilleurs résultats : avoir deux couches d'*average-pooling* dans le réseau à quatre couches donne de meilleurs résultats que lorsqu'on en utilise une seule. Cette opération d'*average-pooling*, en diminuant la taille des caractéristiques, permet de réduire le risque de surapprentissage du modèle. C'est pour cela qu'on observe de meilleurs résultats lorsqu'on en utilise.



Réseau	EAM
64^3	20.1
64^4	17.5
64^2 AP 64	16.6
64 AP 64^2	16.6
64 AP 64 AP 64	16.2
64^3 AP 64	14.5
64^2 AP 64^2	14.4
64 AP 64^3	14.2
64^2 AP 64 AP 64	**13.3**
64 AP 64^2 AP 64	**13.4**
64 AP 64 AP 64^2	13.8
64 AP 64 AP 64 AP 64	**13.3**

FIGURE 5.30 – Évolution de l'erreur absolue moyenne lorsqu'on ajoute des couches d'*average-pooling*. On peut voir que la présence de couches d'*average-pooling* améliore considérablement la qualité des résultats. En effet, si on ajoute des couches d'*average-pooling* au réseau à trois couches, on arrive à atteindre des performances supérieures à celles du réseau à quatre couches. De même, l'ajout de couches d'*average-pooling* au réseau à quatre couches améliore là aussi ses résultats. D'après cette expérience, il semble qu'utiliser plusieurs couches d'*average-pooling* améliore les performances.

Ajout de couches de max-pooling

Dans cette partie, on étudie l'impact de l'ajout de couches de max-pooling. Les couches de *max-pooling* utilisées ont un noyau de taille 2×2 et un pas $(2, 2)$, c'est-à-dire qu'elles sélectionnent la valeur maximale parmi des patchs de caratéristique de taille 2×2 et divisent par deux la hauteur et la longueur des caractéristiques.

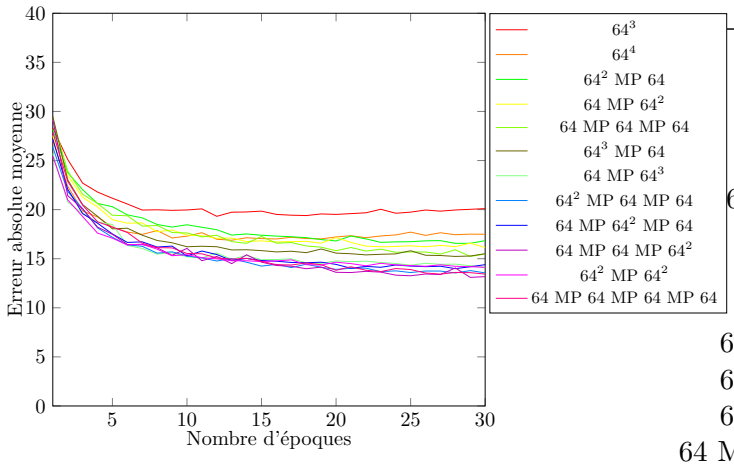
Les résultats sont donnés en Figure 5.31.

On peut observer que l'ajout de couches de max-pooling améliore grandement la qualité de l'apprentissage. Le *max-pooling* permet d'effectuer un sous-échantillonnage de l'image ou des caractéristiques qui en ont été extraites, réduisant ainsi la quantité de paramètres et donc la quantité de calcul dans le réseau et le risque de surapprentissage. Les opérations de *max-pooling* créent aussi une forme d'invariance par translation. Ces propriétés expliquent l'amélioration les résultats de notre estimateur de posture.

Average-pooling, *max-pooling* ou *stride* ?

Dans cette partie, on cherche à déterminer quelle opération est la plus efficace pour améliorer les performances du modèle. Les résultats sont présentés en Tableau 5.5.

On observe que l'utilisation de *pooling* donne systématiquement de meilleures performances que les couches de convolutions avec *stride* (on a une amélioration de l'ordre du

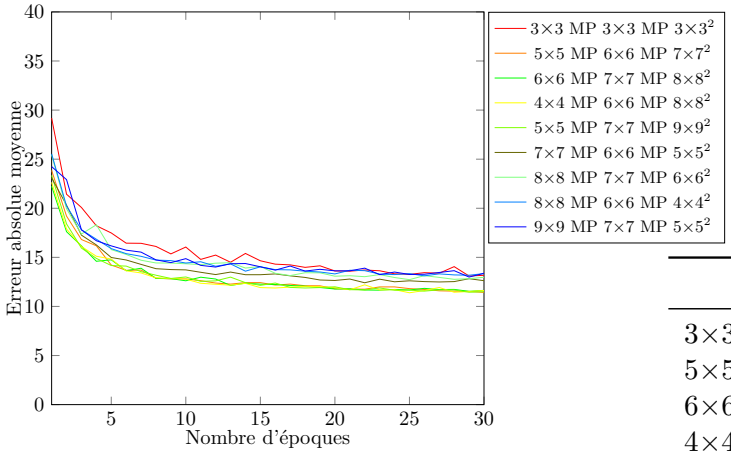


Réseau	EAM
64^3	20.1
64^4	17.5
64^2 MP 64	16.8
64 MP 64^2	16.1
64 MP 64 MP 64	15.5
64^3 MP 64	15.5
64^2 MP 64^2	14.1
64 MP 64^3	14.1
64^2 MP 64 MP 64	13.6
64 MP 64^2 MP 64	14.4
64 MP 64 MP 64^2	**13.2**
64 MP 64 MP 64 MP 64	**13.4**

FIGURE 5.31 – Évolution de l'erreur absolue moyenne lorsqu'on ajoute des couches de *max-pooling*. On observe que la présence de couches de *max-pooling* améliore grandement la qualité des résultats. En effet, si on ajoute des couches de *max-pooling* au réseau à trois couches, on arrive à atteindre des performances supérieures à celles du réseau à quatre couches. De même, l'ajout de couches de *max-pooling* au réseau à quatre couches améliore là aussi ses résultats. D'après cette expérience, il semble qu'utiliser plusieurs couches de *max-pooling* améliore les performances.

Architecture du réseau	Pas de *pooling*	Convolution avec *stride*	Type de pooling *Average*	*Max*
64^2 X 64	20.1	17.4	**16.6**	16.8
64 X 64^2	20.1	16.9	16.6	**16.1**
64 X 64 X 64	20.1	16.8	16.2	**15.5**
64^3 X 64	17.5	15.7	**14.5**	15.5
64^2 X 64^2	17.5	14.8	14.4	**14.1**
64 X 64^3	17.5	14.8	14.2	**14.1**
64^2 X 64 X 64	17.5	14.8	**13.3**	13.6
64 X 64^2 X 64	17.5	14.6	**13.4**	14.4
64 X 64 X 64^2	17.5	14.4	13.8	**13.2**
64 X 64 X 64 X 64	17.5	14.9	**13.3**	13.4

TABLE 5.5 – Comparaison des performances des réseaux de neurones d'estimation de la posture avec *pooling* et *stride*. La présence d'un X signifie que l'on applique une couche de pooling ou que la couche de convolution précédente utilisait un *stride* de taille $(2, 2)$. On observe que le *stride* ou le *pooling* améliorent les performances. Cependant, le *pooling* semble légérement plus efficace que le *stride*, en ayant une erreur absolue moyenne plus faible d'environ 1cm.



Réseau	EAM
3×3 MP 3×3 MP 3×3²	13.2
5×5 MP 6×6 MP 7×7²	**11.4**
6×6 MP 7×7 MP 8×8²	**11.6**
4×4 MP 6×6 MP 8×8²	**11.6**
5×5 MP 7×7 MP 9×9²	**11.5**
7×7 MP 6×6 MP 5×5²	12.6
8×8 MP 7×7 MP 6×6²	13.0
8×8 MP 6×6 MP 4×4²	13.3
9×9 MP 7×7 MP 5×5²	13.4

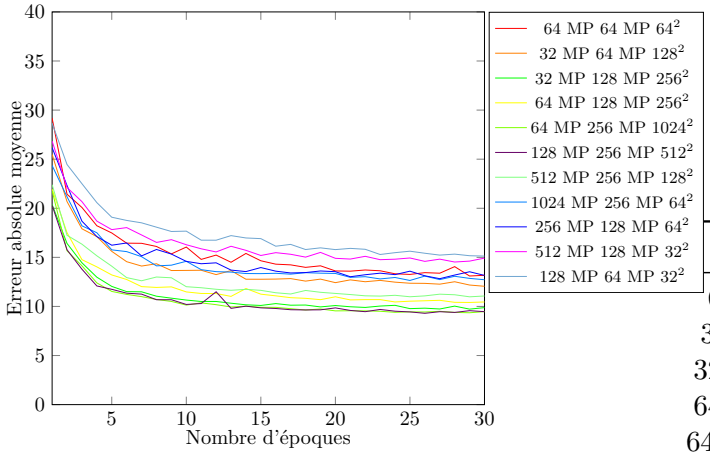
FIGURE 5.32 – Évolution de l'erreur absolue moyenne lorsqu'on ajoute des couches de *max-pooling*. On observe qu'il semble intéressant d'augmenter la taille des filtres après les couches de *max-pooling*. Ceci provoque une importante amélioration des performances du réseau de neurones. En revanche, diminuer la taille des filtres semble légèrement détériorer la qualité des estimations.

centimètre). Cependant, il est difficile de déterminer quelle opération de *pooling* est la plus efficace. En effet, les deux opérations présentent souvent des résultats très proches, et les meilleurs résultats sont alternativement obtenus par l'une ou par l'autre.

5.3.5 Modification des couches convolutives autour des couches de *pooling*

Comme les couches de *pooling* modifient la structure des caractéristiques, on cherche à déterminer si il peut être intéressant de changer les paramètres des couches de convolution lorsqu'on a recours à du *max-pooling*. Nous avons choisi d'utiliser le *max-pooling*, car, grâce à sa structure, il permet d'extraire les caractéristiques les plus importantes, car il récupère les réponses maximales des neurones, plutôt que d'en faire une moyenne.

Tout d'abord, on cherche à déterminer si l'augmentation ou la diminution du nombre de filtres après un max-pooling a une influence sur la qualité de l'apprentissage. Les résultats sont donnés en Figure 5.32. Il semble que l'augmentation du nombre de filtres après une couche de *max-pooling* améliore la performance du réseau, mais la diminution ne semble pas améliorer l'apprentissage. On trouve donc des résultats similaires à ceux obtenus sans couche de *pooling*.



Réseau	EAM
64 MP 64 MP 64^2	14.1
32 MP 64 MP 128^2	12.1
32 MP 128 MP 512^2	9.9
64 MP 128 MP 256^2	10.4
64 MP 256 MP 1024^2	**9.5**
128 MP 256 MP 512^2	**9.5**
512 MP 256 MP 128^2	11.1
1024 MP 256 MP 64^2	12.7
256 MP 128 MP 64^2	13.2
512 MP 128 MP 32^2	14.9
128 MP 64 MP 32^2	15.1

FIGURE 5.33 – Évolution de l'erreur absolue moyenne lorsqu'on modifie le nombre de filtres après *max-pooling*.

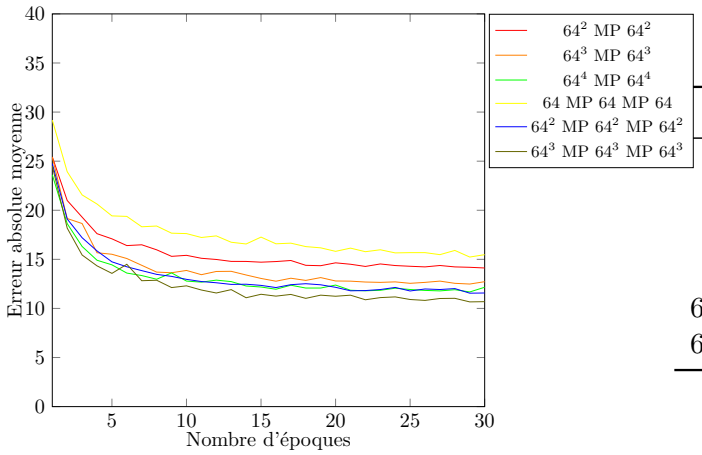
Ensuite, on s'intéresse à la modification de la taille des filtres : on cherche à déterminer si la modification de la taille des filtres après un max-pooling a une influence sur les performances du réseau. Les résultats sont donnés en Figure 5.33. Il semble que l'augmentation de la taille des filtres après une couche de max-pooling améliore la qualité de l'apprentissage, en revanche la diminution de la taille des filtres semble la détériorer. On trouve donc à nouveau des résultats similaires à ceux obtenus sans couche de *pooling*.

Enfin, on cherche à déterminer si la modification du nombre de couches entre les *max-pooling* a une influence sur les performances du réseau. Les résultats sont donnés en Figure 5.34. On observe qu'augmenter le nombre de couches permet d'améliorer les performances du réseau. Ceci est une nouvelle fois conforme aux résultats obtenus sans couche de *max-pooling*.

5.3.6 Dropout

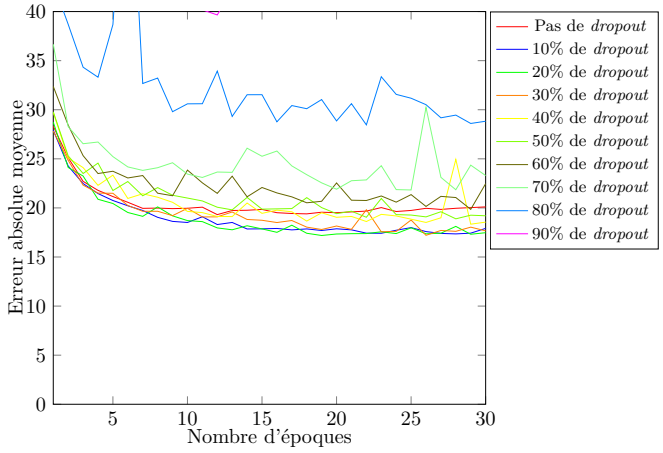
Afin de réduire le surapprentissage constaté sur certains réseaux, on peut recourir au *drop-out*. Cette opération consiste à éteindre aléatoirement certains neurones au cours de l'apprentissage, rendant leur réponse nulle pendant une itération. Ceci permet de réduire la spécialisation des neurones, et donc de maintenir la capacité du réseau de neurones à généraliser. Les résultats sont donnés en Figure 5.35.

On observe que le *dropout* permet de légèrement améliorer les performances du modèle,



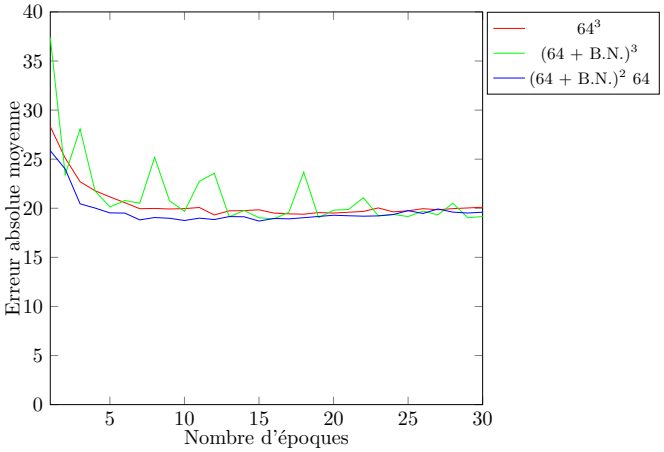
Réseau	EAM
64^2 MP 64^2	14.1
64^3 MP 64^3	12.7
64^4 MP 64^4	12.1
64 MP 64 MP 64	15.5
64^2 MP 64^2 MP 64^2	11.6
64^3 MP 64^3 MP 64^3	**10.7**

FIGURE 5.34 – Évolution de l'erreur absolue moyenne lorsqu'on modifie le nombre de couches après *max-pooling*. On observe qu'augmenter le nombre de couches améliorer les performances de l'estimation de posture.



Taux de *dropout*	EAM
0%	20.1
10%	17.9
20%	**17.5**
30%	**17.7**
40%	18.6
50%	19.2
60%	22.5
70%	23.3
80%	28.8
90%	44.1

FIGURE 5.35 – Évolution de l'erreur absolue moyenne en fonction du taux de *dropout*. L'ajout de dropout semble améliorer l'apprentissage du modèle. En effet, on observe que la qualité des performances augmente lorsqu'on utilise du *dropout* en quantité limitée, c'est-à-dire entre 10 et 30%. Au-delà, l'apprentissage se dégrade, la courbe d'évolution de l'erreur absolue moyenne pour un réseau de neurones avec un taux de *dropout* de 90% étant même hors des limites du graphe.



Réseau	EAM
64^3	20.1
$(64 + \text{B.N.})^3$	**19.1**
$(64 + \text{B.N.})^2\ 64$	19.7

FIGURE 5.36 – Influence des couches de régularisation sur les performances du réseau de neurones d'estimation de posture. On observe que l'utilisation de couches de *Batch Normalization* après chaque couche de convolution améliore légèrement les performances du modèle. Toutefois, ces couches de régularisation provoquent d'importantes oscillations au cours de l'apprentissage.

lorsqu'il est utilisé en faible quantité, avec un taux de 10 à 30%. Cependant, lorsqu'on augmente trop le taux de *dropout*, la qualité de l'apprentissage se dégrade.

Ainsi, un faible taux de *dropout* permet d'améliorer la capacité de généralisation du modèle, mais un taux trop élevé perturbe l'entraînement : les neurones sont trop souvent éteints pour apprendre correctement à résoudre le problème.

5.3.7 Normalisation des données et des caractéristiques

Pour réduire la variabilité, on peut normaliser les données en normalisant les valeurs des pixels des images d'entrée dans différents intervalles, généralement l'intervalle [0 ;1] et plus souvent encore l'intervalle [-1 ;1]. Nou avons choisi de normaliser nos données dans l'intervalle [-1,1] par *Min-Max Scaling*. Il est aussi possible de recourir à des couches de normalisation dans le réseau de neurones. Dans cette section, on étudie l'influence de ces couches.

Pour réduire la variabilité, on fait appel à des couches de normalisation. Les résultats sont présentés en Figure 5.36. On observe qu'utiliser des couches de *Batch Normalization* après chaque couche convolutive améliore légèrement la qualité de l'apprentissage, et semble plus intéressant que lorsqu'on ne l'utilise pas après la dernière couche. On observe cependant d'importantes oscillations au cours de l'apprentissage.

5.4 Architecture du réseau de neurones

Cette partie présente les réseaux de neurones choisis pour réaliser l'estimation de posture. Nous proposons d'abord un réseau optimal, qui est obtenu en sélectionnant les meilleurs résultats des différentes parties de l'étude. Cependant, comme ce réseau de neurones est trop long à entraîner à cause de la complexité de ses couches de convolution, nous proposons ensuite un modèle similaire, mais simplifié, ce qui permet d'accélérer considérablement son entraînement en ne faisant qu'une faible concession sur la qualité de l'estimation de posture.

5.4.1 Meilleur réseau de neurones

Dans cette partie, nous présentons le meilleur modèle de réseau de neurones qui peut être choisi pour réaliser l'estimation de posture sur nos images de profondeur.

Comme tout au long de l'étude, le réseau d'estimation de posture 3D est un réseau de neurones convolutif. Ses entrées sont des images de profondeur de taille 64×64, normalisées par *Min-Max Scaling* dans l'intervalle [-1 ;1], et il produit en sortie un vecteur à 45 dimensions représentant la position 3D des quinze articulations du corps.

Nous avons choisi d'utiliser trois blocs de trois couches convolutives avec un nombre de filtre croissant : le premier bloc contient des couches convolutives et 64 filtres, le second des couches de 256 filtres et le dernier des couches de 1024 filtres. Ce choix suit les résultats obtenu en Section 5.3.2.

Dans l'objectif d'optimiser le nombre d'unités de calcul à installer dans un centre de tri, nous avons décidé que les filtres des couches de convolution sont des filtres de taille 5×5 pour le premier bloc, 6×6 pour le second et 7×7 pour le troisième et dernier bloc (voir Section 5.3.3).

Suite à l'étude menée précédemment, nous avons observé qu'il est important d'utiliser des couches de *pooling* (voir Section 5.3.4) pour améliorer les performances du modèle. Nous avons choisi d'utiliser le *max-pooling*, car, grâce à sa structure, il permet d'extraire les caractéristiques les plus importantes, car il récupère les réponses maximales des neurones. Les blocs de trois couches convolutives sont donc séparés par une couche de *max-pooling*.

L'étude précédente montre qu'utiliser du *drop-out* permet d'améliorer la capacité de généralisation du modèle (voir Section 5.3.6), et donc ses performances sur des données de test. Cependant, au moment de tester notre réseau de neurones, nous nous sommes aperçus que le *drop-out* avait une influence négative sur la qualité de l'estimation. Nos réseaux de neurones n'en contiennent donc pas.

Enfin, la Section 5.3.7 montre qu'il peut être intéressant d'utiliser des couches de régularisation de type *Batch Normalization* après chaque couche de convolution. Notre réseau de neurones contient donc ces couches de normalisation.

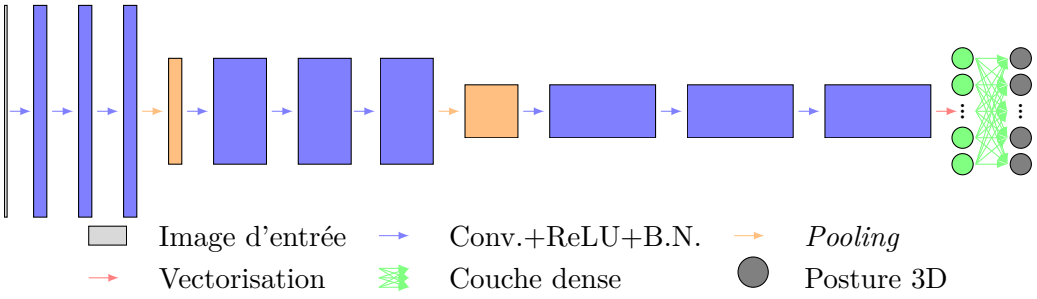


FIGURE 5.37 – Architecture du réseau de neurones optimal d'après l'étude que nous avons menée. Ce réseau de neurones contient trois blocs de trois couches convolutives, où chaque couche de convolution est suivie d'une activation ReLU et d'une couche de *batch normalization*. Les blocs sont séparés par des couches de *max-pooling*. Les couches convolutives contiennent un nombre de filtres qui quadruple après chaque couche de *max-pooling* (64 puis 256 et 1024), et dont la taille est croissante (5×5, puis 6×6 et 7×7). En fin de réseau, on a une couche de vectorisation qui vectorise les caractéristiques, suivie d'une couche dense de quarante-cinq neurones qui permet d'estimer la posture 3D.

Le réseau de neurones optimal est illustré en Figure 5.37 et son architecture détaillée est présentée en Tableau 5.6.

5.4.2 Réseau de neurones utilisé

Cependant, à cause du grand nombre de filtres et de leur taille importante, le modèle présenté précédemment demande beaucoup de temps pour être entraîné. Ainsi, pour le rendre plus facilement déployable en contexte industriel, nous avons choisi de le simplifier pour accélérer son entraînement.

Le réseau d'estimation de posture 3D est toujours un réseau de neurones convolutif. Ses entrées sont des images de profondeur de taille 64×64, normalisées par *Min-Max Scaling* dans l'intervalle [-1 ;1], et il produit en sortie un vecteur à 45 dimensions représentant la position 3D des quinze articulations du corps.

Nous conservons les trois blocs de trois couches convolutives. Cependant, le nombre de filtres de chaque couche reste constant et fixé à 64, mais la taille des filtres est toujours croissante (5×5 pour le premier bloc, 6×6 pour le second et 7×7 pour le troisième), avec une activation ReLU et une couche de *Batch Normalization*, sans *drop-out*. Ces blocs sont toujours séparés par une couche de *max-pooling*.

Ce réseau de neurones est illustré en Figure 5.38 et son architecture détaillée est présentée en Tableau 5.7.

Finalement, nous comparons ces différents réseaux de neurones dans le Tableau 5.8.

Nom	Filtres			Neurones	Sortie	Nombre de paramètres
	Nombre	Taille	Pas			
Image	—	—	—	—	64×64×1	—
Conv1	64	5×5	(1,1)	—	64×64×64	1664
ReLU1	—	—	—	—	64×64×64	0
BatchNorm1	—	—	—	—	64×64×64	256
Conv2	64	5×5	(1,1)	—	64×64×64	102 464
ReLU2	—	—	—	—	64×64×64	0
BatchNorm2	—	—	—	—	64×64×64	256
Conv3	64	5×5	(1,1)	—	64×64×64	102 464
ReLU3	—	—	—	—	64×64×64	0
BatchNorm3	—	—	—	—	64×64×64	256
MaxPooling1	—	2×2	(2,2)	—	32×32×64	0
Conv4	256	6×6	(1,1)	—	32×32×256	590 080
ReLU4	—	—	—	—	32×32×256	0
BatchNorm4	—	—	—	—	32×32×256	1024
Conv5	256	6×6	(1,1)	—	32×32×256	2 359 552
ReLU5	—	—	—	—	32×32×256	0
BatchNorm5	—	—	—	—	32×32×256	1024
Conv6	256	6×6	(1,1)	—	32×32×256	2 359 552
ReLU6	—	—	—	—	32×32×256	0
BatchNorm6	—	—	—	—	32×32×256	1024
MaxPooling2	—	2×2	(2,2)	—	16×16×256	0
Conv7	1024	7×7	(1,1)	—	16×16×1024	12 846 080
ReLU7	—	—	—	—	16×16×1024	0
BatchNorm7	—	—	—	—	16×16×1024	4096
Conv8	1024	7×7	(1,1)	—	16×16×1024	51 381 248
ReLU8	—	—	—	—	16×16×1024	0
BatchNorm8	—	—	—	—	16×16×1024	4096
Conv9	1024	7×7	(1,1)	—	16×16×1024	51 381 248
ReLU9	—	—	—	—	16×16×1024	0
BatchNorm9	—	—	—	—	16×16×1024	4096
Vectorisation	—	—	—	—	262 144	0
Dense	—	—	—	45	45	11 796 525
Posture 3D	—	—	—	—	—	—
Total	—	—	—	—	—	132 937 005

TABLE 5.6 – Architecture détaillée du réseau de neurones optimal. Il contient trois blocs de trois couches convolutives. Chaque couche de convolution est suivie d'une activation ReLU et de *batch normalization*. Les blocs sont séparés par des couches de *max-pooling*. Les couches convolutives contiennent un nombre de filtres qui quadruple après chaque couche de *max-pooling*, de taille croissante. En fin de réseau, une couche de vectorisation vectorise les caractéristiques, suivie d'une couche dense qui permet d'estimer la posture 3D.

Nom	Filtres			Neurones	Sortie	Nombre de paramètres
	Nombre	Taille	Pas			
Image	—	—	—	—	64×64×1	—
Conv1	64	5×5	(1,1)	—	64×64×64	640
ReLU1	—	—	—	—	64×64×64	0
BatchNorm1	—	—	—	—	64×64×64	256
Conv2	64	5×5	(1,1)	—	64×64×64	102 464
ReLU2	—	—	—	—	64×64×64	0
BatchNorm2	—	—	—	—	64×64×64	256
Conv3	64	5×5	(1,1)	—	64×64×64	102 464
ReLU3	—	—	—	—	64×64×64	0
BatchNorm3	—	—	—	—	64×64×64	256
MaxPooling1	—	2×2	(2,2)	—	32×32×64	0
Conv4	64	6×6	(1,1)	—	32×32×64	147 520
ReLU4	—	—	—	—	32×32×64	0
BatchNorm4	—	—	—	—	32×32×64	256
Conv5	64	6×6	(1,1)	—	32×32×64	147 520
ReLU5	—	—	—	—	32×32×64	0
BatchNorm5	—	—	—	—	32×32×64	256
Conv6	64	6×6	(1,1)	—	32×32×64	147 520
ReLU6	—	—	—	—	32×32×64	0
BatchNorm6	—	—	—	—	32×32×64	256
MaxPooling2	—	2×2	(2,2)	—	16×16×64	0
Conv7	64	7×7	(1,1)	—	16×16×64	200 768
ReLU7	—	—	—	—	16×16×64	0
BatchNorm7	—	—	—	—	16×16×64	256
Conv8	64	7×7	(1,1)	—	16×16×64	200 768
ReLU8	—	—	—	—	16×16×64	0
BatchNorm8	—	—	—	—	16×16×64	256
Conv9	64	7×7	(1,1)	—	16×16×64	200 768
ReLU9	—	—	—	—	16×16×64	0
BatchNorm9	—	—	—	—	16×16×64	256
Vectorisation	—	—	—	—	16 384	0
Dense	—	—	—	45	45	737 325
Posture 3D	—	—	—	—	—	—
Total	—	—	—	—	—	1 990 061

TABLE 5.7 – Architecture détaillée du réseau de neurones utilisé. Ce réseau de neurones contient trois blocs de trois couches convolutives, ou chaque couche de convolution contient soixante-quatre filtres de taille croissante (5×5, 6×6 et 7×7) et est suivie d'une activation ReLU et d'une couche de *batch normalization*. Les blocs sont séparés par des couches de *max-pooling*. En fin de réseau, on a une couche de vectorisation qui vectorise les caractéristiques, suivie d'une couche dense de quarante-cinq neurones qui permet d'estimer la posture 3D.

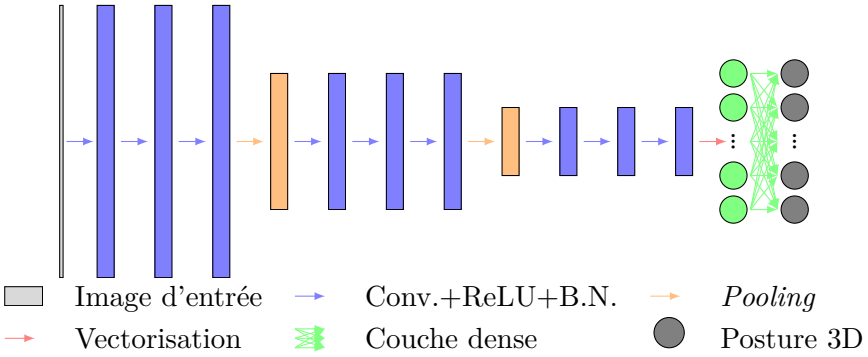


FIGURE 5.38 – Architecture du réseau de neurones utilisé. Ce réseau de neurones contient trois blocs de trois couches convolutives, où chaque couche de convolution contient 64 filtres de taille croissante (5×5, 6×6 et 7×7) et est suivie d'une activation ReLU et d'une couche de *Batch Normalization*. Les blocs sont séparés par des couches de *max-pooling*. En fin de réseau, on a une couche de vectorisation qui vectorise les caractéristiques, suivie d'une couche dense de quarante-cinq neurones qui permet d'estimer la posture 3D.

Nous y avons ajouté d'autres versions simplifiées construites suite à l'étude. Les seuls paramètres qui varient à l'intérieur de ces modèles sont le nombre des filtres des couches convolutives (croissant de 64 à 1024, ou fixé à 64) et la taille des filtres (croissante de 5×5 à 7×7 ou fixé à 4×4). On observe que les modèles avec un grand nombre de filtres de convolution ont un nombre très important de paramètres, avec 132 937 005 paramètres contre 1 990 061. Ils sont donc plus coûteux car ils nécessitent plus d'espace mémoire sur la carte graphique. De plus, le réglage de ces paramètres provoque une augmentation importante du temps d'apprentissage, pour un faible bénéfice au niveau de l'erreur absolue moyenne, qui décroît de 6.5mm à 5.1mm. C'est pour cette raison que nous avons choisi de réduire le nombre de filtres : l'entraînement est considérablement accéléré, diminuant de 41h à 1h50min, en échange d'une légère augmentation de l'erreur, de 5.1mm à 6.5mm, soit une augmentation de 1.4mm.

5.5 Estimation de posture sur des images de synthèse

Dans cette partie, nous nous intéressons à l'évaluation de la qualité de l'estimation de posture réalisée par le réseau de neurones choisi (voir Section 5.4.2).

Pour rappel, ce réseau de neurones est constitué de trois blocs de trois couches convolutives, séparés par des couches de *max-pooling*, et suivis d'une couche dense qui réalise l'estimation de posture 3D. Le nombre de filtres de chaque couche de convolution est fixé à 64, mais la taille des filtres est toujours croissante (5×5 pour le premier bloc, 6×6 pour

Nom	Filtres convolutifs Nombre	Taille	Paramètres	Durée d'entraînement	EAM
Optimal	64 256 1024	5×5 6×6 7×7	132 937 005	41h00min	5.1
Simplifié 1	64 256 1024	4×4 4×4 4×4	52 056 813	15h45min	5.4
Simplifié 2	64 64 64	4×4 4×4 4×4	1 265 517	1h40min	7.3
Utilisé	64 64 64	5×5 6×6 7×7	1 990 061	1h50min	6.5

TABLE 5.8 – Comparaison entre les différents réseaux construits suite à l'étude. Chacun des réseaux suit la même architecture, et les paramètres qui varient sont ceux des couches convolutives. Ils sont donnés par bloc de trois couches convolutives.

le second et 7×7 pour le troisième), avec une activation ReLU et une couche de *Batch Normalization*, sans *drop-out*.

Les résultats de l'estimation de posture 3D sur des images de synthèse sont présentés en Figure 5.39 et Figure 5.40, puis en Figure 5.41 sur une séquence d'images de profondeur de synthèse consécutives. On observe que le modèle est capable de donner des estimations de posture fiables, et les figures confirment visuellement les mesures d'écart effectuées précédemment.

Comme nous souhaitons calculer des indicateurs basés sur les angulations des articulations de la partie supérieure du corps, nous cherchons à déterminer si l'estimation de posture réalisée par notre modèle nous permet d'obtenir des indicateurs fiables. Le procédé de calcul des angulations et des indicateurs est donné en Annexe 2.2. Les écarts d'angulations sont données en Figure 5.42 et en Tableau 5.9. De plus, les erreurs de prédiction au niveau des indicateurs sont données en Tableau 5.10. Nous observons que le modèle nous permet de calculer des angulations qui sont généralement cohérentes avec la posture de l'opérateur : l'erreur moyenne d'estimation est très faible (en moyenne 2.27°), mais il y a cependant quelques estimations aberrantes, comme le montre une erreur de mesure de l'angulation de l'élévation de l'épaule gauche de 129.69°. On observe que les rares valeurs aberrantes sont provoquées par des auto-occultations de l'opérateur. Toutefois, on peut considérer que l'estimation de posture fournie par le modèle nous permet de calculer les indicateurs de manière fiable avec plus de 92.5% de réussite en moyenne.

Ainsi, cette étude nous a permis de mettre en place un réseau de neurones léger et efficace

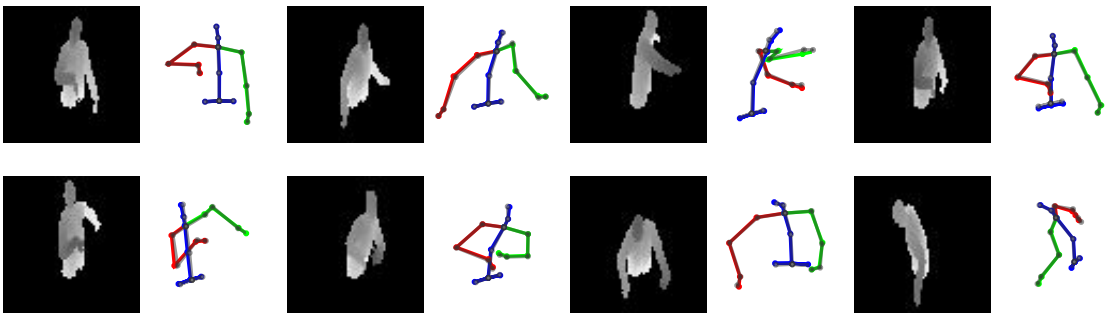


FIGURE 5.39 – Exemple de résultats obtenus par le réseau de neurones utilisé pour notre projet sur des images de synthèse. La posture estimée est affichée en couleur (tronc en bleu, bras gauche en rouge, bras droit en vert), alors que la vérité terrain est donnée en gris. On observe que les estimations de posture sont souvent très proches des vérités terrains, ce qui confirme visuellement la valeur d'erreur absolue moyenne obtenue par ce réseau. Sur l'image en bas à droite, le bras en vert est invisible sur l'image de profondeur, ce qui provoque un écart plus important.

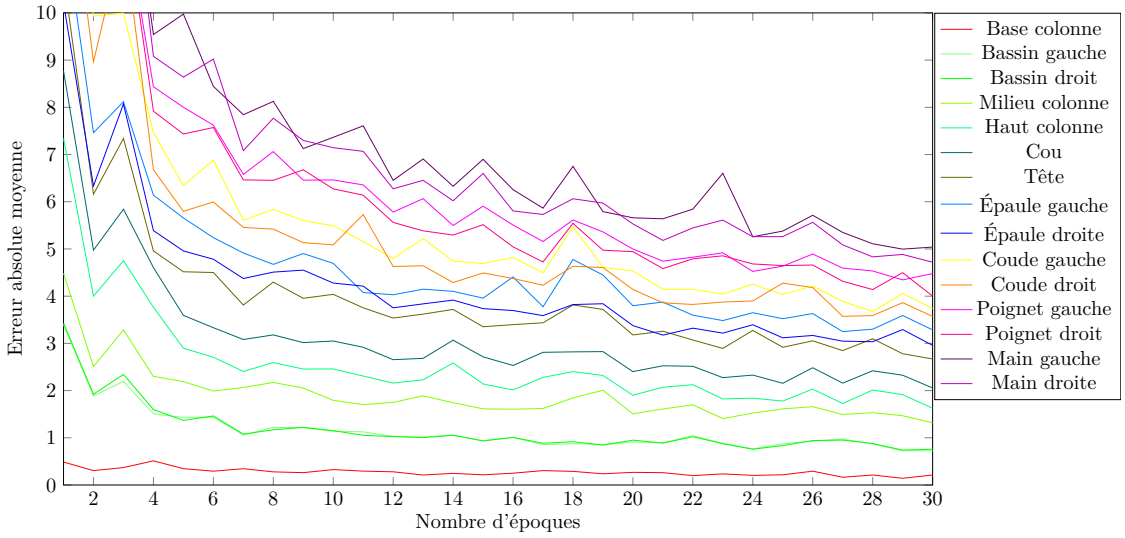


FIGURE 5.40 – Évolution de l'erreur absolue moyenne par articulation pendant l'entraînement du réseau final. On remarque que les erreurs sont très faibles, mais le classement des articulations avec le plus d'écart reste le même : on a d'abord les mains, puis les poignets, les coudes et les épaules, puis la tête, le cou et les articulations du tronc.

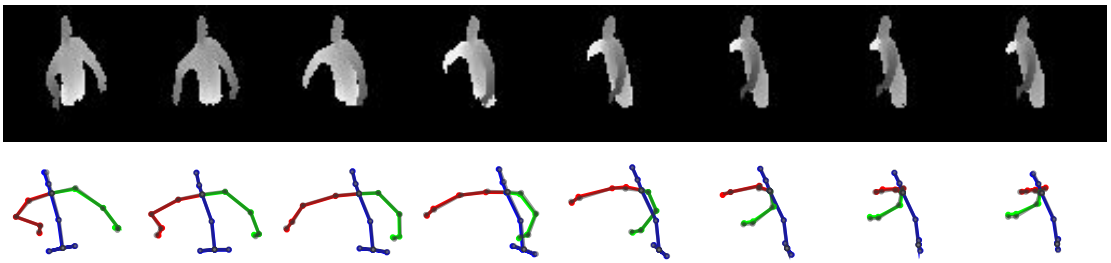
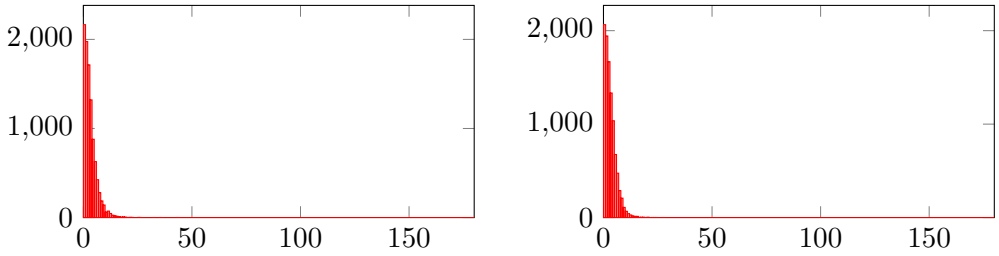


FIGURE 5.41 – Exemple de résultats obtenus par le réseau de neurones utilisé pour notre projet sur des images de synthèse. La posture estimée est affichée en couleur (tronc en bleu, bras gauche en rouge, bras droit en vert), alors que la vérité terrain est donnée en gris. On observe que les estimations de posture suivent les vérités terrains tout au long de la séquence.

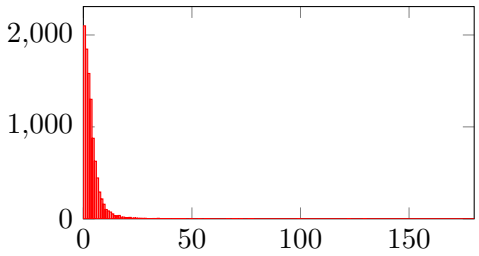
Articulation	Angulation	Erreur				Écart
		Min.	Médiane	Moyenne	Max.	Type
Cou	Flexion	0.00	2.85	3.43	22.05	2.68
Coude gauche	Extension	0.00	2.48	3.18	43.31	2.86
Coude droit	Extension	0.00	2.56	3.16	27.12	2.60
Buste	Rotation	0.00	1.21	1.50	9.51	1.24
	Flexion vers l'avant	0.00	0.38	0.46	2.78	0.37
	Flexion sur le côté	0.00	0.40	0.51	5.53	0.44
Épaule gauche	Élévation	0.00	**2.64**	**3.75**	**129.69**	4.78
	Rotation	0.00	1.43	1.89	20.54	1.77
Épaule droite	Élévation	0.00	2.36	3.14	31.67	2.92
	Rotation	0.00	1.33	1.70	13.38	1.47
Toutes	Toutes	0.00	1.42	2.27	129.69	2.72

TABLE 5.9 – Tableau des erreurs d'angulation après estimation de posture par notre modèle, mesurées en degrés. On observe que les erreurs d'angulation sont généralement très faibles, avec une médiane qui ne dépasse pas 2.64° et une moyenne inférieure à 3.75°. Il y a cependant quelques angulations aberrantes, avec un très grand écart, par exemple une erreur de 130° sur une estimation de l'élévation de l'épaule gauche. Après analyse, il s'agit d'un cas où le bras gauche est occulté par le corps de l'opérateur.

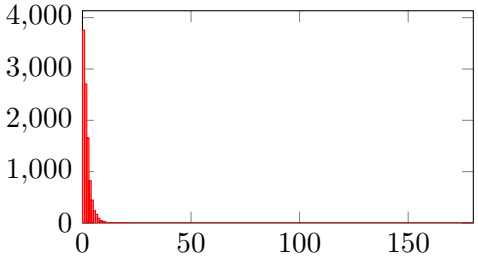


(a) Flexion du coude gauche.

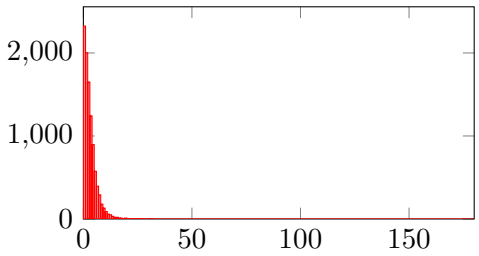
(b) Flexion du coude droit.



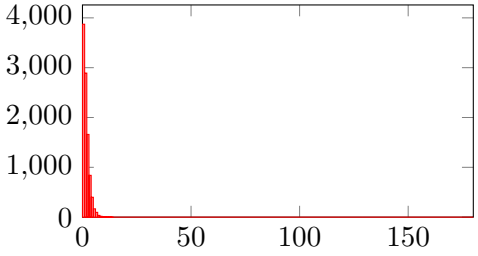
(c) Rotation de l'épaule gauche.



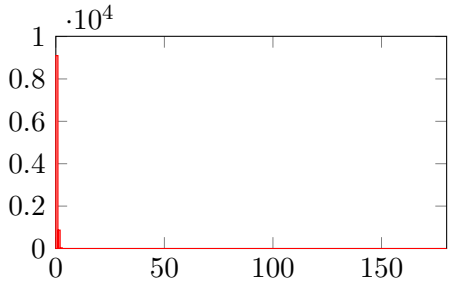
(d) Élévation de l'épaule gauche.



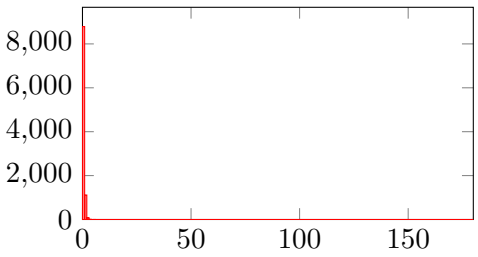
(e) Rotation de l'épaule droite.



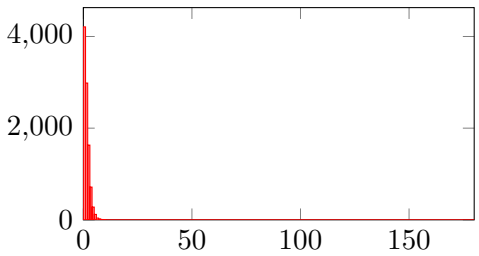
(f) Élévation de l'épaule droite.



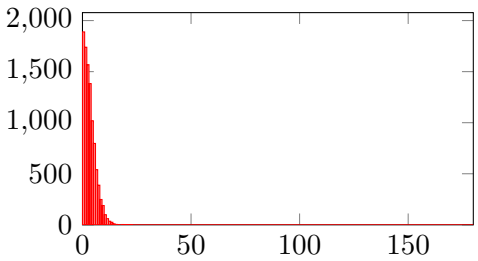
(g) Flexion vers l'avant du buste.



(h) Flexion sur le côté du buste.



(i) Rotation du buste.



(j) Flexion du cou.

FIGURE 5.42 – Histogrammes des écarts d'angulation calculés après estimation de posture. Tous les écarts sont mesurés en degrés. Ces histogrammes montrent que l'erreur d'angulation est en général très faible. On observe que les erreurs les plus faibles sont obtenues pour les angulations du buste, et les écarts les plus importants se situent au niveau des coudes, des épaules et du cou.

Articulation	Angulation	Zones		Zones d'écart	
		Correctes	Erronnées	1	2
Cou	Flexion	100.0%	0.0%	—	—
Coude gauche	Extension	96.0%	4.0%	4.0%	—
Coude droit	Extension	96.2%	3.8%	3.8%	—
Buste	Rotation	96.2%	3.8%	3.8%	—
	Flexion vers l'avant	99.1%	0.9%	0.9%	—
	Flexion sur le côté	**92.5**%	7.5%	7.5%	—
Épaule gauche	Élévation	95.5%	4.5%	4.2%	0.3%
	Rotation	98.4%	1.6%	1.6%	—
Épaule droite	Élévation	95.5%	4.5%	4.4%	0.1%
	Rotation	96.7%	3.3%	3.3%	—

TABLE 5.10 – Différences de zones d'angulation après estimation de la posture par le réseau de neurones sélectionné. On observe que les zones d'angulation sont pour une très écrasante majorité correctes avec au minimum 92.5% de zones correctement calculées. Cependant, on remarque que certaines zones semblent plus difficiles que d'autres. En analysant les résultats, on remarque que l'erreur peut avoir deux origines : une mauvaise estimation de la position des articulations, qui entraîne un calcul d'angulation erroné, comme cela peut-être le cas pour les coudes ou les épaules, ou bien une taille de zone réduite, dans le cas de la flexion du buste sur le côté.

pour l'estimation de posture 3D, qui nous permet de calculer des indicateurs posturaux fidèles à la réalité sur des images de profondeur de synthèse.

5.6 Utilisation du réseau sur les images réelles

Nous tentons maintenant d'appliquer le réseau de neurones d'estimation de posture 3D sur les images réelles. Comme nous ne disposons pas d'annotation pour ces images, nous utilisons un modèle entraîné sur les images de profondeur de synthèse.

Ce réseau de neurones est le même que celui utilisé précédemment : il contient trois blocs de trois couches convolutives, séparés par des couches de *max-pooling*, et qui sont suivis par une couche dense qui réalise l'estimation de posture 3D. Le nombre de filtres de chaque couche de convolution est fixé à 64, mais la taille des filtres est toujours croissante (5×5 pour le premier bloc, 6×6 pour le second et 7×7 pour le troisième), avec une activation ReLU et une couche de *Batch Normalization*, sans *drop-out*.

Les résultats de l'estimation de posture 3D sur les images réelles sont proposés en

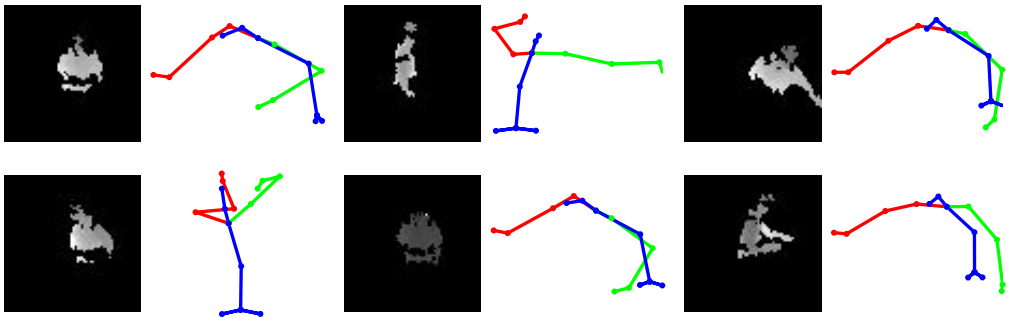


FIGURE 5.43 – Test du réseau d'estimation de posture sur des images réelles. La posture estimée est affichée en couleur (tronc en bleu, bras gauche en rouge, bras droit en vert), mais on ne dispose pas de vérité terrain. On observe que les estimations de posture semblent visuellement très éloignées de la posture réelle de d'opérateur. Ainsi, le réseau de neurones entraîné directement sur les images de synthèse n'est pas capable de donner des résultats intéressants sur les images réelles.

Figure 5.43. On voit que le modèle ne semble pas en mesure d'estimer correctement la posture de l'opérateur. En effet, même si nous ne possédons pas d'annotation pour évaluer un écart entre la vérité et l'estimation, on observe que les postures estimées ne semblent visuellement pas en accord avec la posture de l'opérateur.

Ainsi, il n'est donc pas possible d'utiliser un modèle directement entraîné sur des images de profondeur de synthèse pour réaliser l'estimation de posture sur des images réelles acquises en contexte industriel. Nous avons donc besoin d'une étape supplémentaire, qui nous permet de transformer les images de synthèse en images réalistes, plus proches des images réelles : c'est l'objet du chapitre suivant.

En résumé

Dans ce chapitre, nous présentons la première étape de la construction d'un réseau de neurones léger et efficace pour l'estimation de posture 3D sur des images de profondeur.

Comme nous ne possédons pas d'annotation pour entraîner un modèle sur les images de profondeur réelles, nous construisons des images de profondeur de synthèse, pour lesquelles nous pouvons construire une annotation de posture sous la forme d'un squelette cinématique. Le procédé de construction de cet ensemble de données est présenté en Section 5.1.

En Section 5.2, nous décrivons les méthodes de l'état de l'art qui permettent de réaliser une estimation de posture 2D ou 3D sur des images.

Ensuite, nous proposons en Section 5.3 une étude sur l'influence de l'architecture du modèle sur la qualité de l'estimation de posture. Nous mesurons ainsi l'impact de la structure des couches convolutives (nombre et taille des filtres), de leur nombre, de la présence ou de l'absence de couches de *pooling*, de régularisation et de *drop-out*.

L'étude menée en Section 5.3 nous permet de construire un réseau de neurones pour l'estimation de posture 3D sur des images de profondeur de synthèse. Son architecture est détaillée en Section 5.4. Nous proposons de plus dans cette partie une simplification du réseau de neurones pour rendre son entraînement plus rapide, et ainsi permettre un déploiement plus facile.

Nous montrons l'efficacité du modèle sélectionné en Section 5.5. Nous vérifions que l'étude menée nous a permis de construire un réseau de neurones léger et efficace, c'est-à-dire dont les estimations de posture sur les images de profondeur de synthèse sont proches de la vérité terrain. Cette partie permet de confirmer visuellement les mesures effectuées précédemment, et de montrer que la posture estimée par ce réseau de neurones peut être utilisée pour calculer les indicateurs ergonomiques.

Enfin, nous indiquons cependant en Section 5.6 que le modèle ne permet pas d'effectuer l'estimation de posture 3D sur des images réelles. En effet, les résultats proposés par le réseau de neurones semblent visuellement très différents de la posture affichée par l'opérateur. Ceci nous motive donc à construire un dispositif capable de transformer les images de profondeur de synthèse en images réalistes, plus proches des images de profondeur réelles.

Chapitre 6

Traduction d'image à image

Sommaire

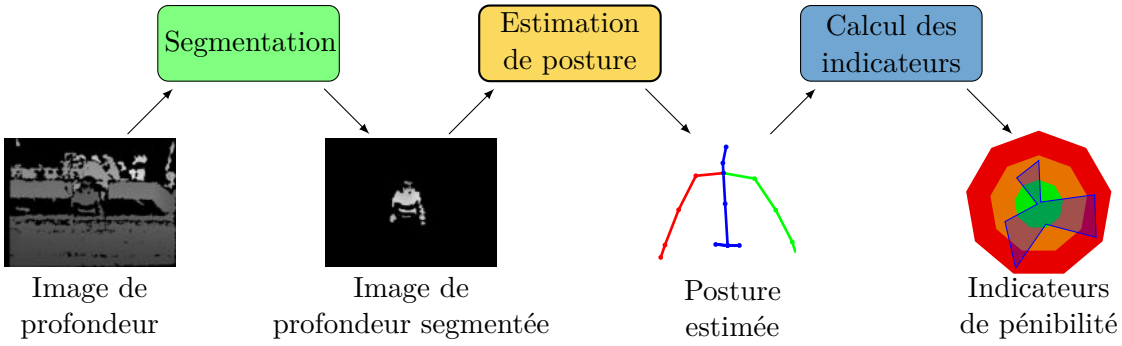


FIGURE 6.1 – Notre pipeline pour le calcul des indicateurs de pénibilité à partir des images de profondeur. Dans ce chapitre, nous considérons le premier module terminé (en vert), et nous nous concentrons sur le second module (en jaune), qui nous permettra de construire une estimation de la posture de l'opérateur humain à partir de l'image de profondeur segmentée.

Dans ce chapitre, nous travaillons toujours sur le deuxième module de notre pipeline (voir Figure 6.1). Suite à l'étude que nous avons menée en Chapitre 5, nous cherchons à transformer nos images de profondeur de synthèse en images de profondeur plus réalistes, c'est-à-dire qui prennent mieux en compte les caractéristiques de nos images de profondeur réelles, telles que les dégradations provoquées par les vêtements réfléchissants portés par l'opérateur ou l'incertitude du capteur. Ainsi, nous proposons d'utiliser un module de traduction d'image à image nous permettant de transformer automatiquement les images de profondeur de synthèse en images de profondeur réalistes. Ces images de profondeur réalistes sont ensuites utilisées pour entraîner le réseau de neurones d'estimation de posture.

Tout d'abord, nous proposons d'utiliser un CycleGAN (voir Section 6.3). Cet outil de traduction d'image à image est capable d'apprendre sur des ensembles de données non appairés, c'est-à-dire que les images fournies en entrée du modèle n'ont pas besoin d'être associées à une image attendue en sortie pendant l'apprentissage.

Nous utilisons ensuite les images réalistes, obtenues à l'aide des images de profondeur de synthèse, ainsi que leur annotation de posture associée, pour entraîner un modèle d'estimation de posture fonctionnant sur les images de profondeur réelles (voir Section 6.4). Ce réseau de neurones d'estimation de posture est construit d'après l'étude décrite en Chapitre 5.

Plusieurs expériences, présentées en Section 6.5, montrent que l'approche que nous proposons est intéressante et permet d'améliorer significativement les performances du réseau de neurones, et donc la qualité de l'estimation de posture réalisée sur des images de profondeur réelles, lorsqu'on compare ses résultats à ceux d'un modèle entraîné directement sur les images de synthèse (comme effectué en Section 5.6).

6.1 Rappel du contexte et des motivations

Notre objectif, décrit précédemment en Chapitre 2, est de concevoir un système qui aiderait à identifier les situations à risque, dans le but de prévenir l'apparition de troubles musculo-squelettiques chez un opérateur humain, en analysant la posture de celui-ci pour signaler les angulations nuisibles. Poussés par notre contexte industriel, nous devons réaliser ces opérations sous plusieurs contraintes (présentées en Section 1.4), dont celles qui influent le plus sur la qualité des données acquises sont les suivantes :

— la surveillance doit être non invasive, pour éviter de troubler le travail des opérateurs ;
— les conditions d'éclairage ne peuvent pas être contrôlées : le capteur peut faire face à une fenêtre ou être dans un coin sombre de l'établissement ;
— l'anonymat des travailleurs doit être préservé pour respecter leur vie privée et éviter les utilisations abusives des données collectées ;
— les opérateurs peuvent porter des vêtements de travail réfléchissants.

Pour toutes ces raisons, nous avons choisi de présenter ce problème comme une estimation de la posture humaine sur des images de profondeur (voir Section 1.5).

L'estimation de la posture humaine, qui consiste à localiser des points clés (articulations) ou des parties anatomiques du corps humain, a toujours été une tâche très difficile en vision par ordinateur, aussi bien en 2D qu'en 3D (voir Sections 3.3 et 5.2). La plupart des travaux récents se concentrent sur l'estimation de la posture d'une personne seule en 2D, comme par exemple ceux de [Newell et al., 2016], et utilisent différentes architectures de réseaux de neurones convolutifs appliqués sur des images en couleur.

Un autre axe de travail aborde le problème dans l'optique d'estimer simultanément la posture de plusieurs personnes, toujours sur des images en couleur. Ces méthodes plus récentes utilisent souvent un modèle "*top-down*" [Papandreou et al., 2017] où une détection des humains est d'abord effectuée, puis un algorithme d'estimation de posture pour une seule personne est appliqué à chaque détection. Cependant, les méthodes les plus efficaces reposent sur des approches "*bottom-up*" [Cao et al., 2017, Insafutdinov et al., 2016]. Ces méthodes utilisent des blocs ResNet [He et al., 2016] (voir Section 3.4.2) ou une cascade de réseaux de neurones convolutifs pour détecter d'abord les articulations, puis les faire correspondre pour effectuer une estimation complète de la posture.

L'estimation de la posture 3D par réseaux de neurones à partir d'images uniques reste un défi à ce jour, en raison du manque de données étiquetées disponibles et de l'ambiguïté de l'obtention d'informations 3D à partir d'images uniques. Grâce à des méthodes d'apprentissage profond, la posture 3D est souvent déduite de la posture estimée en 2D, comme dans les travaux de [Tome et al., 2017], car les réseaux de neurones qui apprennent sur des ensembles de données acquis dans un contexte de capture de mouvement [Ionescu et al., 2013, Sigal et al., 2010] ne peuvent pas être utilisés sur des données naturelles [Li and Chan, 2014, Zhou et al., 2016a].

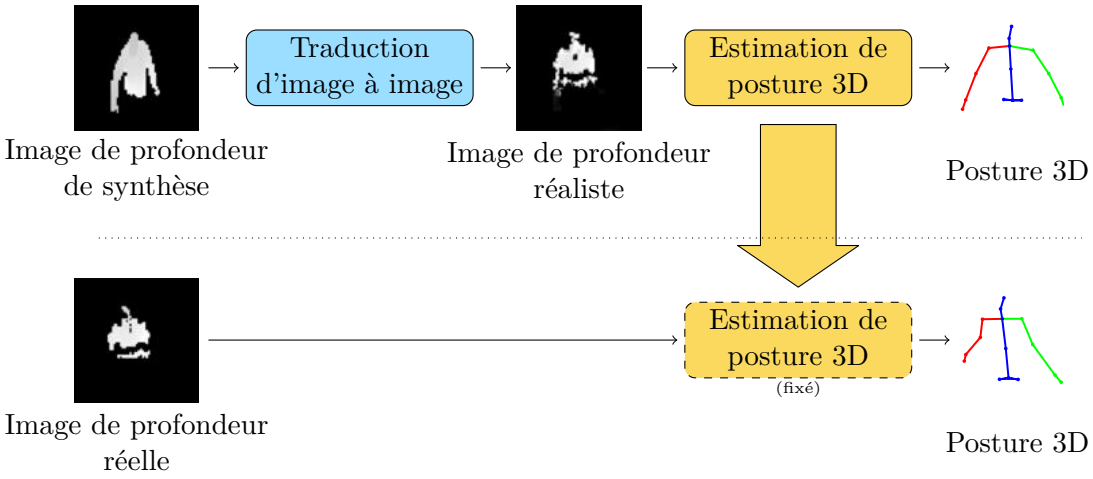


FIGURE 6.2 – Notre procédé d'estimation de posture par apprentissage semi-supervisé. Un premier module de traduction d'image à image (en bleu), transforme des images de profondeur de synthèse en images de profondeur réalistes, c'est-à-dire visuellement plus proches des images de profondeur réelles acquises dans un contexte industriel. Ces images de pronfondeur réalistes, associées à la posture des images de profondeur de synthèse dont elles sont la transformation, sont ensuite utilisées par un second module d'estimation de posture (en jaune) pendant la phase d'entraînement de celui-ci. Une fois cet estimateur de posture entraîné, il peut être utilisé de manière fiable pour estimer la posture sur des images de profondeur réelles.

Pour s'attaquer à ce problème, une approche non supervisée basée sur l'adaptation de domaine a très récemment été proposée par [Zhang et al., 2019]. Les auteurs entraînent tout d'abord un estimateur de posture 2D/3D sur des images de profondeur de synthèse et un module de segmentation à l'aide de la posture 2D. En utilisant l'adaptation de domaine, les auteurs utilisent ensuite ces deux composants pour entraîner un estimateur de posture 2D/3D sur des images en couleur.

Même si les capteurs RVB-D sont utilisés par le grand public depuis de nombreuses années (Microsoft a publié la première version du capteur Kinect en 2010), seuls quelques travaux de recherche portaient sur la manière d'inférer des informations en utilisant uniquement des images de profondeur. En particulier, la plupart des travaux basés sur l'analyse de l'humain utilisent la segmentation et l'estimation de la posture intégrées dans le Kinect [Shotton et al., 2011], qui suppose implicitement que l'utilisateur se tient devant le capteur. Comme adopté précédemment par [Banerjee et al., 2012, Chou et al., 2018] dans le cadre des hôpitaux pour la reconnaissance des actions et la détection de chutes, notre approche se concentre sur l'utilisation des seules informations de profondeur du capteur, sans tenir compte de l'information sur les couleurs, afin de respecter la vie privée des opérateurs humains.

Cela implique d'estimer la posture humaine à partir d'images de profondeur non étique-

tées et bruitées. Pour ce faire, nous présentons dans cette partie un processus d'estimation de posture semi-supervisé, décrit en Figure 6.2. C'est un véritable défi, car, comme vu précédemment (voir Section 5.6), si l'on entraîne le modèle d'estimation de posture avec des images de profondeur de synthèse structurellement trop éloignées des images de profondeur réelles, le réseau de neurones ne sera pas capable de réaliser des prédictions pertinentes sur ces images réelles.

Notre procédé s'appuie sur l'approche CycleGAN [Zhu et al., 2017], un modèle particulier de réseaux antagonistes génératifs, capable de préserver les propriétés géométriques de l'image tout en changeant sa texture. Nous proposons d'utiliser ce modèle sur nos images de profondeur de synthèse (voir Chapitre 5), pour obtenir des images réalistes, qui sont ensuite utilisées pour entraîner le réseau de neurones d'estimation de posture. Afin de s'assurer de la pertinence de notre approche, nous réalisons différentes expériences qui nous permettent d'évaluer nos résultats.

6.2 Vers un apprentissage semi-supervisé

Dans cette partie, nous présentons des concepts qui ont servi d'inspiration pour les travaux proposés dans ce chapitre. Tout d'abord, nous présentons un autre type d'apprentissage : l'apprentissage semi-supervisé. C'est un apprentissage qui combine l'utilisation de données annotées et de données non étiquetées. Ensuite, nous détaillons quelques approches de traduction d'image à image, qui permettent de modifier le style d'une image, tout en conservant son contenu.

L'apprentissage semi-supervisé est une approche de l'apprentissage automatique qui combine l'utilisation de données annotées avec celle de données non annotées pendant l'entraînement. C'est donc un apprentissage qui se situe entre l'apprentissage supervisé et l'apprentissage non supervisé.

Les données non annotées, lorsqu'elles sont utilisées en parallèle de données annotées, peuvent grandement améliorer les performances des modèles entraînés [Blum and Mitchell, 1998].

L'annotation des données est souvent coûteuse, aussi bien financièrement qu'en temps humain : l'étiquetage des données demande l'intervention d'un expert pour annoter ou vérifier la qualité des annotations. Ces coûts peuvent donc rendre impossible la construction de grands jeux de données annotées, alors que l'acquisition de données brutes est en général peu coûteuse. Dans ces cas là, l'apprentissage semi-supervisé peut s'avérer pratique.

Ces méthodes sont intéressantes dans le cadre de cette thèse, car peu de données annotées nous sont accessibles. De plus, si l'on souhaite mettre en place en conditions réelles le système développé au cours de cette thèse, il n'est pas envisageable d'annoter des données pour chaque dispositif installé. Ainsi, il est possible que les apprentissages, probablement distincts pour chaque centre voire poste de tri en raison des différences d'environnement,

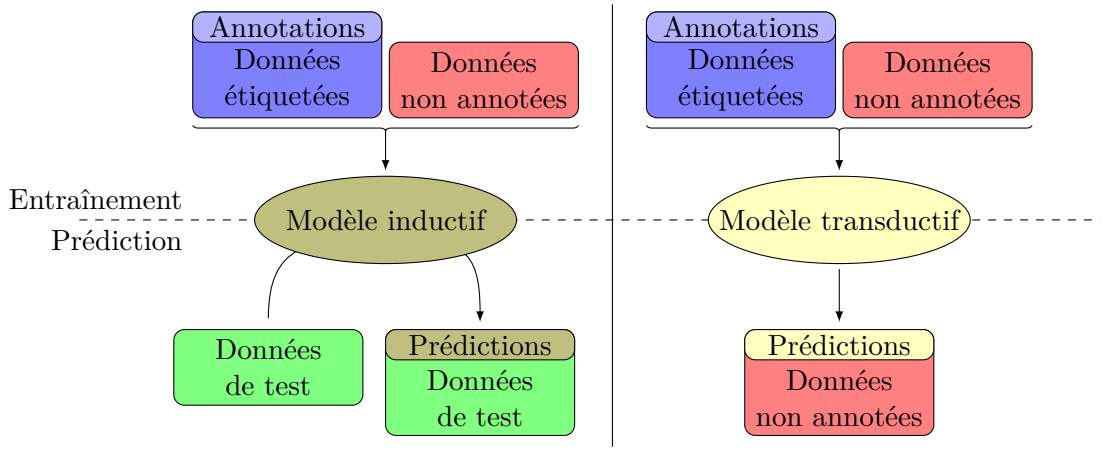


FIGURE 6.3 – Illustration de la différence entre l'apprentissage inductif et l'apprentissage transductif : l'apprentissage inductif cherche à construire un modèle général pouvant être utilisé sur des données inconnues au moment de l'apprentissage, tandis que l'apprentissage transductif cherche uniquement à donner des prédictions pour les données non annotées.

se basent sur un faible nombre de données annotées, réutilisées pour chaque apprentissage, et un grand nombre de données non annotées, propres à chaque centre ou poste de tri.

L'apprentissage semi-supervisé peut aussi bien être inductif que transductif [Zhu, 2006] :

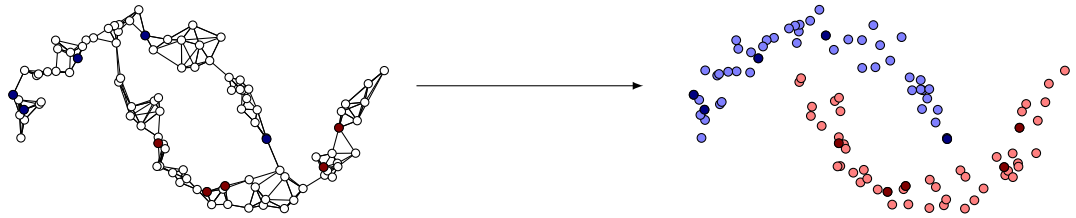
— l'apprentissage est **inductif** lorsque l'on cherche à construire un modèle général à partir des échantillons individuels, aussi bien annotés que non étiquetés. L'objectif est donc de construire un modèle qui fonctionne sur de nouvelles données, qui ne sont pas utilisées pendant la phase d'entraînement ;

— l'apprentissage est **transductif** lorsque l'on raisonne des échantillons individuels annotés vers les échantillons non étiquetés. Les échantillons non annotés sont donc des données de test en même temps que des données d'entraînement. L'objectif de l'apprentissage transductif est de maximiser les performances sur les données non annotées. Le modèle n'est pas fait pour être utilisé sur des données inconnues, car on ne cherche pas à le doter d'une capacité de généralisation.

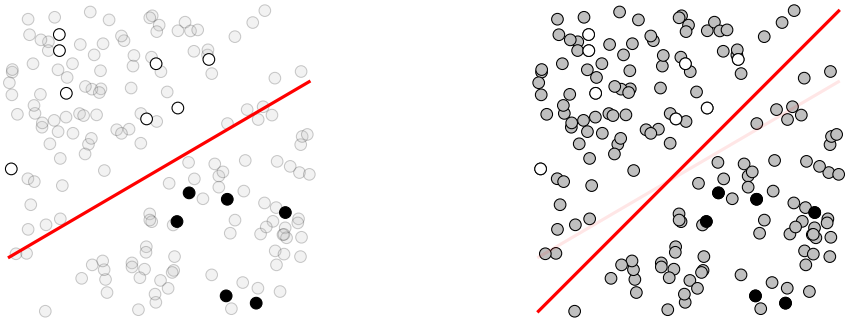
Une illustration de la différence entre ces deux sortes d'apprentissage est donnée en Figure 6.3.

6.2.1 Méthodes d'apprentissage semi-supervisé

La principale difficulté de l'apprentissage semi-supervisé consiste à trouver un moyen d'utiliser conjointement les données annotées et les données non annotées. Il existe plusieurs types d'approches, qui sont présentées ici.



(a) Exemple de classification par propagation dans un graphe. À gauche, on a les données annotées (cercles colorés) et non annotées (blanc), auxquelles on a ajouté le graphe correspondant au cinq plus proches voisins. À droite, on a la classification réalisée par propagation dans le graphe.



Séparatrice sans utilisation des données non annotées (SVM)

Séparatrice avec utilisation des données non annotées (TSVM)

(b) Exemple de machine à vecteurs de suport transductive. Les données annotées sont représentées en noir et blanc, tandis que les données non annotées sont affichées en gris. La séparatrice optimale est donnée en rouge. On remarque que l'utilisation des données non annotées (à droite) modifie considérablement la séparatrice obtenue lorsqu'on on ne se sert que des données étiquetées (à gauche).

FIGURE 6.4 – Exemples de méthodes d'apprentissage semi-supervisé. On y retrouve la propagation dans un graphe (a) et la machine à vecteurs de support transductive (b).

Machine à vecteurs de support transductive

Dans le cadre de l'apprentissage supervisé, les SVM maximisent la marge à l'aide de données annotées. Ils ont été étendus au cadre semi-supervisé par les machines à vecteurs de support transductives (abrégé TSVM pour *transductive support vector machine*). Les TSVM maximisent alors la marge sur l'ensemble des données, en proposant, pour les données non annotées, l'étiquette la plus favorable, par minimisation de la probabilité d'erreur. Les machines à vecteurs de support transductives améliorent la solution supervisée lorsque la séparatrice se trouve dans une zone de faible densité [Joachims, 1999]. Un exemple illustratif est donné en Figure 6.4b.

Propagation dans un graphe

Des approches basées sur les graphes sont aussi utilisées en apprentissage semi-supervisé. Elles s'appuient sur une matrice de voisinage construite à partir de la distance entre les échantillons pour construire le graphe. Un algorithme itératif est ensuite appliqué pour propager les annotations des données étiquetées vers celles qui ne le sont pas [Zhou et al., 2004]. Ces approches reposent sur une hypothèse de densité qui suppose que deux points proches et situés dans une même zone de forte densité doivent avoir une étiquette semblable. Un exemple de classification par propagation dans un graphe est donné en Figure 6.4a. Ces méthodes sont transductives : on ne peut étiqueter que des données appartenant au graphe. Si on ajoute des données, il faut reconstruire le graphe et réappliquer l'algorithme de propagation.

Régularisation d'un modèle supervisé

Un modèle construit par apprentissage supervisé peut être instable quand le jeu de données d'entraînement est de taille réduite. Utiliser les données non annotées peut s'avérer utile pour régulariser le modèle. Par exemple, si l'on suppose que la séparation entre deux classes doit intervenir dans une zone de faible densité, on peut régulariser le modèle à l'aide de l'entropie [Grandvalet and Bengio, 2006].

Des méthodes de réduction de dimension peuvent être utilisées afin de permettre de régulariser les solutions obtenues : les données non annotées permettent une réduction de dimension, qui, comme dans le cadre de l'apprentissage non supervisé (voir Section 3.1.2), peut par exemple être réalisée par analyse en composantes principales [Pearson, 1901]. L'enjeu est d'arriver à représenter les données dans un espace de petite dimension avec une perte minimale d'information. Une fois la réduction de dimension effectuée, les méthodes d'apprentissage supervisé usuelles sont utilisées sur les données projetées dans l'espace de dimension réduite.

Méthodes génératives

Des méthodes génératives permettent de faire de l'apprentissage semi-supervisé en s'appuyant sur des modèles génératifs. Un modèle génératif est un modèle statistique qui utilise la probabilité conjointe pour effectuer ses prédictions. Les méthodes génératives utilisées pour l'apprentissage semi-supervisé sont proches de celles utilisées en apprentissage non supervisé : on a par exemple les mélanges gaussiens (abrégé GMM de l'anglais *Gaussian Mixture Model*) [Ma et al., 2017]. L'estimation des paramètres du modèle est généralement réalisée à l'aide de l'algorithme d'espérance-maximisation [Dempster et al., 1977].

Autres approches

L'auto-apprentissage (en anglais *self-training*) consiste à itérativement entraîner le modèle grâce aux données annotées, puis à ajouter des données non annotées à l'ensemble de l'entraînement en les étiquetant grâce aux prédictions du modèle [Rosenberg et al., 2005]. L'entraînement continue jusqu'à ce qu'il n'y ait plus de donnée non étiquetée. L'auto-entraînement est donc une approche heuristique basée sur l'apprentissage supervisé. Les variantes de cette approche se concentrent sur les critères qui permettent de sélectionner les données à ajouter : par exemple, on peut ajouter toutes les données, en les pondérant selon un indice de confiance, ou bien n'ajouter que les données avec la plus forte confiance [Triguero et al., 2015]. L'intérêt de ces approches est qu'elles sont relativement simples à mettre en place. Cependant, comme des erreurs de prédiction sur les données non annotées peuvent fortement dégrader les performances du modèle, il faut être en mesure de retirer des données si l'on constate une dégradation des résultats.

Le co-apprentissage, proposé en 1998 par Avrim Blum et Tom Mitchell [Blum and Mitchell, 1998], est une approche d'apprentissage proche de l'auto-entraînement. Elle utilise plusieurs modèles entraînés sur des ensembles différents de caractéristiques, idéalement disjoints et indépendants [Krogel and Scheffer, 2004], qui servent à construire des annotations les uns pour les autres. Le co-entraînement construit premièrement des modèles distincts, en utilisant les données annotées. Les prédictions les plus fiables de chaque modèle sont ensuite utilisées de manière itérative pour contruire des données d'entraînement supplémentaires.

6.2.2 Modifier le style d'une image

Lorsqu'on souhaite modifier le style d'une image tout en conservant son contenu, on est confronté à un problème de traduction d'image à image. La traduction d'image à image, de l'anglais *image-to-image translation*, est un type de problèmes de graphisme et de vision par ordinateur, où l'objectif est d'apprendre une correspondance entre un ensemble d'images données en entrée, et un ensemble d'images attendues en sortie. L'objectif est donc de transformer une ou plusieurs images d'un domaine en images ayant le style ou les caractéristiques d'une ou plusieurs images d'un autre domaine, tout en conservant leur contenu. Un exemple de traduction d'image à image est présenté en Figure 6.5.

La restauration ou la reconstruction d'images peuvent être vues comme des problèmes de traduction d'image à image. Dans cet état de l'art, nous nous concentrons sur les méthodes qui cherchent à modifier le style d'une image pour la transformer en une autre. Ainsi, nous n'abordons pas ici les approches de traduction d'image à image pour la super-résolution, l'*inpainting*, le débruitage ou le défloutage. Nous ne présentons pas non plus les méthodes de traduction de vidéo à vidéo telles que Vid2Vid [Wang et al., 2018c, Wang et al., 2019b].

La traduction d'image à image fait partie du champ de recherche plus large de l'adap-

Image originale Images traduites

FIGURE 6.5 – Exemple de traduction d'une image en images d'un autre style. L'image source est donnée à gauche et les images traduites sont à droite. On observe que même si le style change, le contenu est préservé (lunettes, visage, habits, etc.).

tation de domaine [Ganin et al., 2016]. L'adaptation de domaine consiste à adapter un système d'apprentissage d'un domaine source vers un domaine cible. Dans cette partie, nous présentons uniquement les approches qui cherchent à transformer le style d'une image.

On distingue trois types d'approches pour réaliser la traduction d'une image en une autre :

- les méthodes basées sur un apprentissage à l'aide de données appariées : ces méthodes utilisent des ensembles de données où chaque donnée d'entrée peut être associée à une donnée de sortie. Ces approches ont donc recours à de l'apprentissage supervisé (voir Section 3.1.1) ;
- les méthodes basées sur un apprentissage avec des données non appariées : cette fois-ci, les approches proposées ont accès à des ensembles de données couvrant le domaine d'entrée et le domaine de sortie, mais il n'y a pas d'appariement possible entre les données d'entrée et les données de sortie. On est donc dans un cas d'apprentissage non supervisé (voir Section 3.1.2) ;
- les méthodes basées sur le transfert de style. Ici, on va se servir d'un algorithme capable d'extraire des informations sur le style d'une image, et modifier itérativement le style de l'image d'entrée jusqu'à ce que son style soit celui d'une image de référence.

Dans cet état de l'art, nous commençons par présenter les approches de transfert de style. Nous décrivons ensuite les méthodes basées sur un apprentissage à partir de données appariées. Nous terminons en présentant les approches utilisant des données non appariées.

Transfert de style

Le transfert de style consiste à récupérer le style d'une image de référence pour l'appliquer sur une image source, que l'on souhaite transformer. De nombreuses approches et méthodes ont été proposées, et elles sont résumées dans une revue récente de la littérature [Jing et al., 2019].

Dans cette partie, nous présentons le principe de la traduction d'image à image par

transfert de style. Les travaux sur lesquels nous nous appuyons pour écrire cette partie ont été proposés par [Gatys et al., 2016]. Ainsi, cette section n'a pas pour objet de décrire les différentes méthodes de l'état de l'art, mais plutôt d'expliquer le fonctionnement du transfert de style.

Le transfert de style est une technique qui ne nécessite pas d'apprentissage direct, c'est-à-dire qu'on ne va pas nécessairement apprendre à un modèle à transformer des images d'un premier domaine pour qu'elles prennent le style d'images d'un second domaine. Cependant, l'apprentissage peut intervenir au niveau du modèle utilisé pour extraire le style des images. Ce modèle d'extraction de style est en général un modèle qui a été entraîné pour une autre tâche. Par exemple, un réseau de neurones utilisé pour comparer les styles est VGG [Simonyan and Zisserman, 2015], un réseau de neurones initialement construit et entraîné pour la classification d'images.

Le problème du transfert de style est donc un problème d'optimisation : on va chercher à minimiser l'écart entre le style de l'image de référence R et de l'image transformée T, tout en préservant le contenu de l'image source O. Résoudre le problème du transfert de style revient donc à minimiser la fonction de perte suivante :

$$\mathcal{L}(O, R, T) = \alpha \mathcal{L}_{contenu}(O, T) + \beta \mathcal{L}_{style}(R, T) \tag{6.1}$$

où :

— $\mathcal{L}_{contenu}(O, T)$ mesure la différence de contenu entre O et T ;

— $\mathcal{L}_{style}(R, T)$ mesure la différence de style entre R et T ;

— α et β sont des coefficients d'influence relative entre le contenu et le style.

Ainsi, la principale difficulté consiste à définir les fonctions $\mathcal{L}_{contenu}(O, T)$ et $\mathcal{L}_{style}(R, T)$.

L'approche proposée par [Gatys et al., 2016] part d'un constat simple : le développement des réseaux de neurones convolutifs a permis de construire des modèles de vision par ordinateur très puissants, dont les représentations de caractéristiques apprises, de plus en plus complexes à mesure que la profondeur dans le réseau augmente, peuvent être utilisées pour manipuler indépendamment le contenu et le style d'images. Ainsi, les auteurs proposent d'extraire des caractéristiques construites par un réseau de neurones afin de mesurer la différence de contenu entre l'image source et l'image transformée et la différence de style entre l'image transformée et l'image référence.

Le principe de l'approche est illustré en Figure 6.6. Tout d'abord, pour mesurer la différence de contenu entre l'image source et l'image transformée, les auteurs proposent de comparer les réponses d'une même couche c d'un réseau de neurones lorsqu'il traite l'image source et l'image transformée. Si l'on note n_c le nombre de filtres d'une couche de convolution et s_c la taille d'une carte de caractéristiques construites par un filtre, la réponse R^c de la couche c est une matrice $M \in \mathbb{R}^{n_c \times s_c}$, où R^c_{ij} est l'activation de la position j du $i^{\text{ème}}$ filtre de la couche c. Si l'on note respectivement O^c et T^c la réponse de

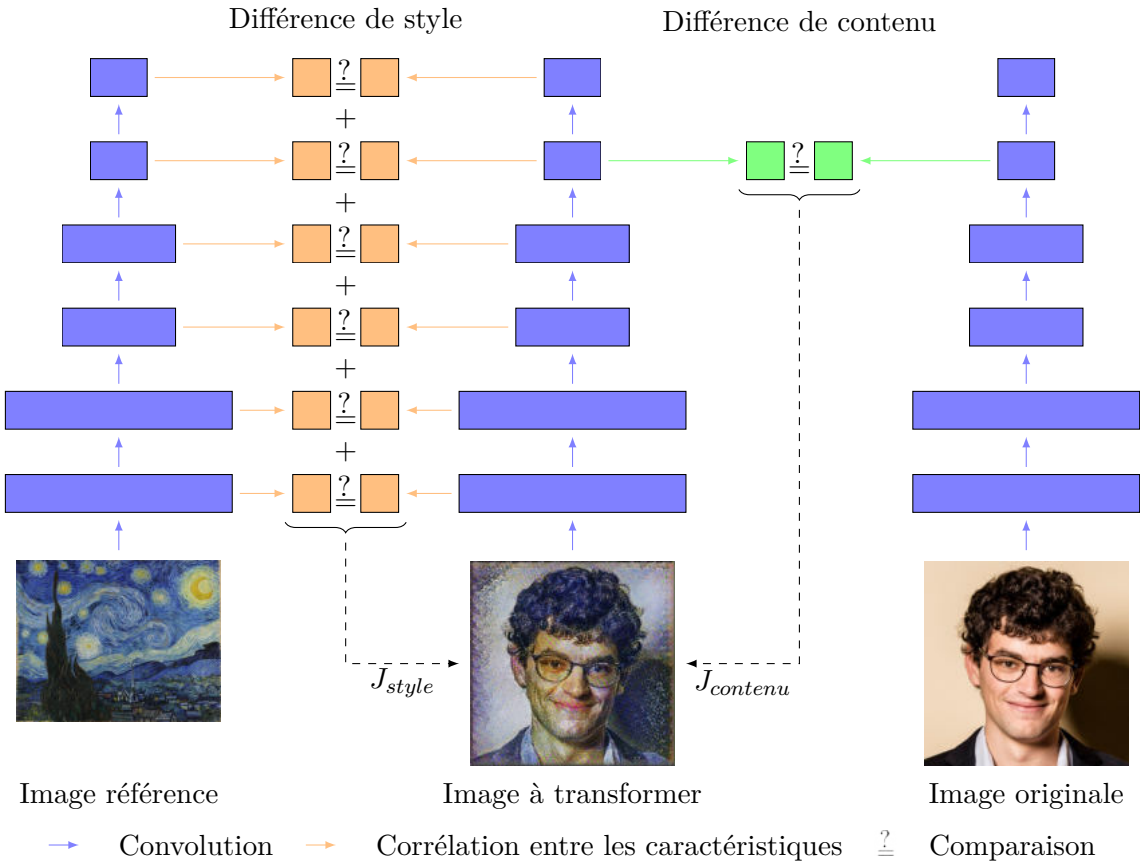


FIGURE 6.6 – Fonctionnement du transfert de style. À l'aide des couches convolutives d'un réseau de neurones, on peut extraire l'information de style de l'image originale (à gauche) et l'information de contenu d'une image référence (à droite). La différence de contenu est obtenue en mesurant les différences de réponse d'une même couche pour l'image originale et l'image à transformer, alors que la différence de style est obtenue en mesurant les différences de corrélation entre les caractéristiques des différentes couches du réseau pour l'image référence et l'image à transformer. Une descente de gradient permet de faire converger l'image à transformer vers l'image optimale, qui minimise la somme de ces deux différences.

la couche c du réseau de neurones aux images originale et transformée, [Gatys et al., 2016] définissent la différence de contenu pour une couche c comme :

$$\mathcal{L}_{contenu}(O,T,l) = \frac{1}{2}\sum_{i,j}\left(T_{ij}^{c} - O_{ij}^{c}\right)^2 \tag{6.2}$$

Les auteurs conseillent d'utiliser des couches profondes pour comparer le contenu.

Ensuite, pour comparer le style entre deux images, les auteurs suggèrent de mesurer la corrélation entre les activations des différents filtres d'une même couche c, et de comparer les valeurs de corrélation obtenues pour l'image référence et l'image transformée. Ces

corrélations peuvent être mesurées à l'aide de la matrice de Gram $G^c \in \mathbb{R}^{n_c \times n_c}$, ou G^c_{ij} est le produit scalaire entre les cartes de caractéristique i et j de la couche c. Si l'on utilise la corrélation de plusieurs couches, on obtient une représentation qui capture l'information de style sans tenir compte du contenu. Si l'on note respectivement O^c et T^c la matrice des corrélations des activations de la couche c du réseau de neurones en réponse aux images originale et transformée, la contribution de la couche c à la différence de style est :

$$E_c(R,T) = \frac{1}{4n_c^2 m_c^2} \sum_{i,j} \left(T^c_{ij} - O^c_{ij}\right)^2 \tag{6.3}$$

et la différence de style totale est :

$$\mathcal{L}_{style}(R,T) = \sum_{c=0}^{C} w_c E_c(R,T) \tag{6.4}$$

avec :

— w_c les coefficients de contribution pour chaque couche à la différence globale ;

— C le nombre total de couches ;

$\mathcal{L}_{contenu}$ et $\mathcal{L}_{style}$ étant dérivables par rapport à T, on peut effectuer une descente de gradient pour faire converger T vers la fonction qui minimise $\mathcal{L}$. L'image T^* obtenue est la traduction de l'image originale par transfert de style.

Des travaux proposés par [Johnson et al., 2016] ont considérablement accéléré la méthode en utilisant un réseau de neurones transformateur d'images, entraîné pour résoudre le problème d'optimisation, en reprenant le principe d'utilisation d'un réseau auxiliaire pour mesurer la différence de style et de contenu des images générées par le modèle transformateur. L'approche a été améliorée par [Gatys et al., 2017] en introduisant un contrôle spatial, colorimétrique et sur l'échelle pour l'application du transfert de style.

Traduction d'image à image avec données appariées

D'autres travaux sur le problème de traduction d'image à image s'appuient sur des ensembles de données appariées, c'est-à-dire pour lesquels les images d'un premier domaine source sont associées aux images du second domaine. Ceci permet en particulier d'entraîner efficacement un modèle de manière supervisée.

L'approche la plus populaire est Pix2Pix [Isola et al., 2017]. Ce modèle est générique et permet de résoudre plusieurs problèmes de traduction d'image à image tels que la transformation d'une vue aérienne en plan, la transformation d'annotations en scène ou la colorisation d'images.

Le modèle est constitué de réseaux antagonistes génératifs conditionnés [Mirza and Osindero, 2014]. En effet, la donnée en entrée du générateur est l'image du domaine source que l'on souhaite transformer en image du domaine cible, et le discriminateur doit distinguer

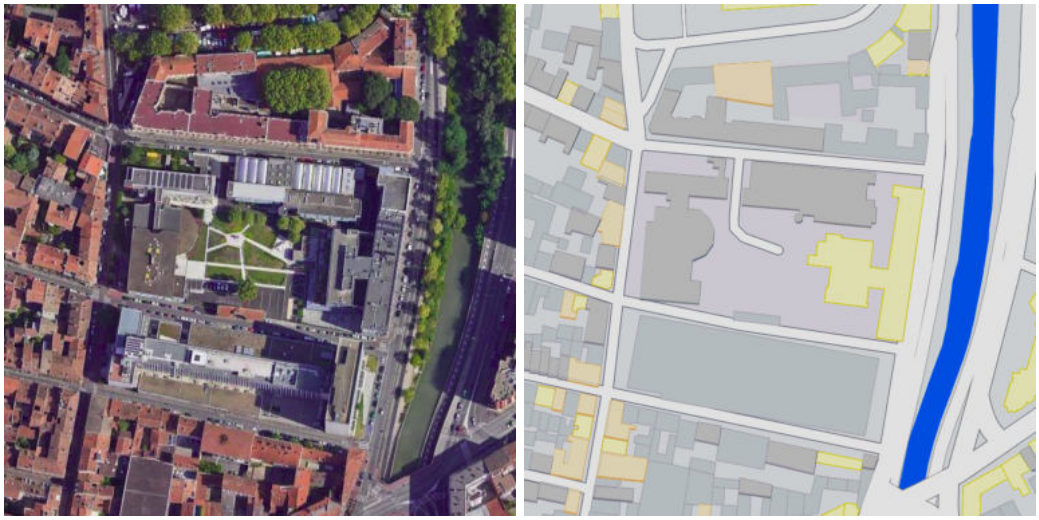


FIGURE 6.7 – Exemple d'images appariées. Lorsqu'on cherche à transformer des vues aériennes (à gauche) en plans (à droite), on peut utiliser des appariements d'images pour entraîner le modèle de traduction d'image à image.

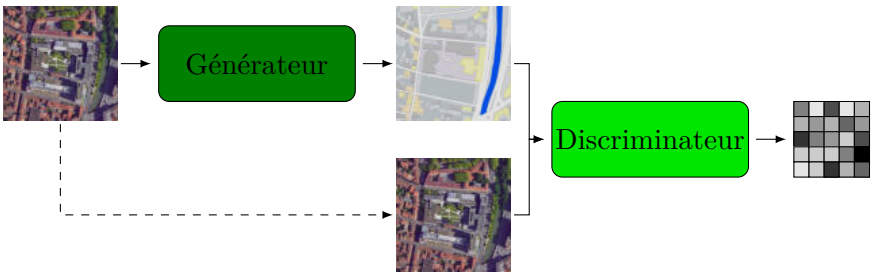


FIGURE 6.8 – Fonctionnement du modèle Pix2Pix. Le générateur prend en entrée un image du domaine source, qu'il transforme en image du domaine cible. L'image utilisée par le générateur et celle reconstruite sont ensuite analysées par le discriminateur qui détermine si la paire est originale ou bien si l'image du domaine cible a été générée. Le discriminateur est un patchGAN : au lieu de donner une unique valeur, il traite la paire par patchs d'image, et propose une probabilité pour chaque patch.

des paires d'images (image source/image cible associée ou image source/image générée), comme illustré en Figure 6.8. Sans cette utilisation conditionnée du générateur, il serait impossible de traduire directement une image en une autre : en plus de générer une image, le générateur doit être conditionné pour que cette image reste cohérente avec celle donnée en entrée. Afin que l'information géométrique de bas niveau soit conservée, le générateur est un modèle inspiré de U-net [Ronneberger et al., 2015], car la présence de *skip connections* permet de transporter directement l'information des premières couches vers celles des dernières couches. Le discriminateur est un PatchGAN, c'est-à-dire un discriminateur qui va travailler sur des patchs de l'image et proposer en sortie une carte de probabilité

plutôt qu'une unique valeur. Les auteurs montrent que le PatchGAN peut être vu comme un genre de fonction de perte sur le style, car il modélise l'image comme un champ aléatoire de Markov avec une indépendance entre les pixels dont la distance est supérieure à la taille du patch.

Cette approche a été utilisée par [Zhang et al., 2017b] pour la coloration de dessins, où le générateur détermine la coloration à donner au dessin à partir de caractéristiques extraites de VGG [Simonyan and Zisserman, 2015]. Dans [Sangkloy et al., 2017], les auteurs reprennent Pix2Pix en ajoutant des termes de régularisation pour la colorisation supervisée de dessin. En particulier, on mesure la différence entre l'image générée et l'image cible, ainsi que la différence de contenu, à la manière de [Gatys et al., 2016]. En parallèle, les auteurs de [Wang et al., 2018a] proposent d'améliorer le système en introduisant une fonction de perte perceptuelle, mesurant la différence de style, basée sur l'extraction de caractéristiques à partir des couches convolutives du discriminateur, similairement aux travaux de transfert de style de [Gatys et al., 2016]. À la différence de Pix2Pix, le discriminateur prend alors uniquement en entrée l'image construite par le générateur. Dans [Wang et al., 2018b], les auteurs améliorent Pix2Pix pour la génération d'image en haute résolution, à l'aide de générateurs multi-échelle imbriqués et de leur discriminateurs associés. [AlBahar and Huang, 2019] reprend Pix2Pix et introduit une approche où la génération d'image est guidée : l'architecture du générateur est adaptée pour utiliser un "guide" (par exemple, un morceau de texture), fourni en entrée avec l'image source, qui permet d'améliorer la qualité des images construites par le générateur.

Une autre approche de traduction d'image à image utilisant les images appariées a été proposée par [Wang and Gupta, 2016]. Cette fois, l'objectif est de générer en cascade une image d'un premier domaine à partir d'un vecteur de bruit, puis de transformer cette image en image du second domaine (voir Figure 6.9). La traduction d'image à image intervient donc dans la deuxième partie du modèle. Plus précisément, les auteurs construisent dans un premier temps des cartes de normales de surface à l'aide d'un sous-modèle appelé Structure-GAN. Ces cartes de normales de surface sont ensuite transformées en images d'intérieur grâce à un sous-modèle appelé Style-GAN. On se concentre ici seulement sur le sous-modèle Style-GAN, qui effectue la tâche de traduction d'image à image, et dont l'architecture et le fonctionnement sont très différents du modèle Pix2Pix. À la différence du générateur de Pix2Pix, le générateur du Style-GAN, bien que conditionné en entrée par une image, se sert aussi d'un vecteur de bruit uniforme. Après une phase où la carte des normales de surface et le vecteur de bruit uniforme sont traités indépendamment, le générateur a une structure d'encodeur-décodeur. L'architecture du discriminateur est aussi différente : ce n'est plus un PatchGAN, mais un discriminateur "global", qui donne en sortie une unique valeur.

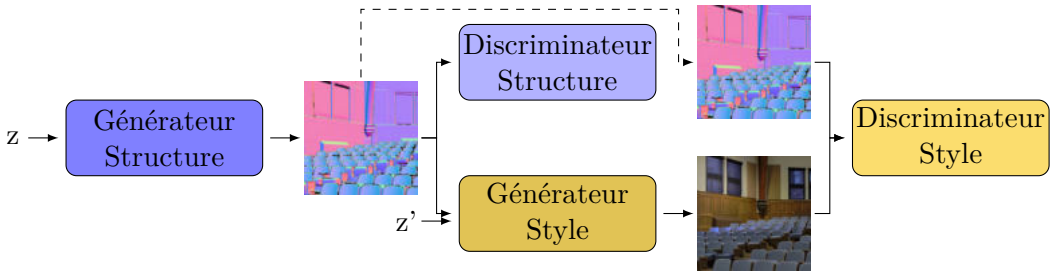


FIGURE 6.9 – Le modèle de S^2-GAN proposé par [Wang and Gupta, 2016]. En nuances de bleu, le sous-modèle Structure-GAN et en nuances de jaune, le sous-modèle Style-GAN, qui réalise l'étape de traduction d'image à image. En entrée, le générateur du Structure-GAN prend un vecteur de bruit uniforme z, qu'il utilise pour construire une carte de normales de surface, qui est analysée par le discriminateur du Structure-GAN. Cette carte de normales de surface est ensuite utilisée comme conditionnement du générateur du Style-GAN, couplé à un nouveau vecteur de bruit uniforme z'. Ceci permet de construire l'image de scène intérieure associée à l'image de normales de surfaces. Enfin, le discriminateur de Style analyse la paire formée par l'image de normales de surface générée par le générateur du Structure-GAN et l'image de scène intérieure construite par le générateur du Style-GAN. On a donc besoin d'images appariées pour entraîner le Style-GAN.

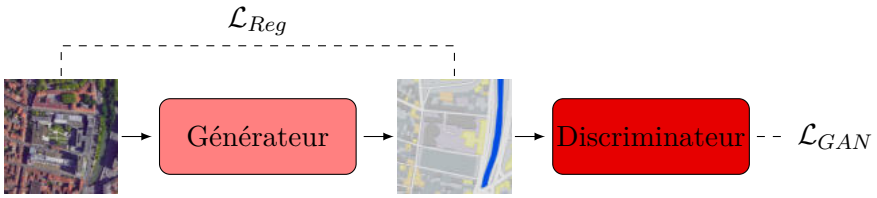


FIGURE 6.10 – Illustration du fonctionnement du SimGAN [Shrivastava et al., 2017]. En plus de la fonction de perte $\mathcal{L}_{GAN}$ calculée par le GAN, les auteurs proposent d'utiliser un critère de régularisation $\mathcal{L}_{Reg}$ pour comparer directement l'image fournie en entrée à l'image donnée en sortie.

Traduction d'image à image avec données non appariées

Bien que les approches basées sur une traduction d'image à image où le modèle est entraîné à l'aide de données appariées produisent des résultats prometteurs, elles présentent l'inconvénient de nécessiter un appariement, souvent difficile à mettre en place en pratique. Par conséquent, d'autres travaux se sont intéressés à une approche moins supervisée du problème, où les images de l'ensemble de données source ne sont pas appariées aux images du domaine cible. Les approches proposées nécessitent alors de nouveaux composants de régularisation, pour s'assurer de la pertinence des images générées.

Pour ce faire, les auteurs de [Shrivastava et al., 2017] proposent l'approche SimGAN. Cette approche consiste à utiliser un modèle de réseaux génératifs antagonistes (GAN)

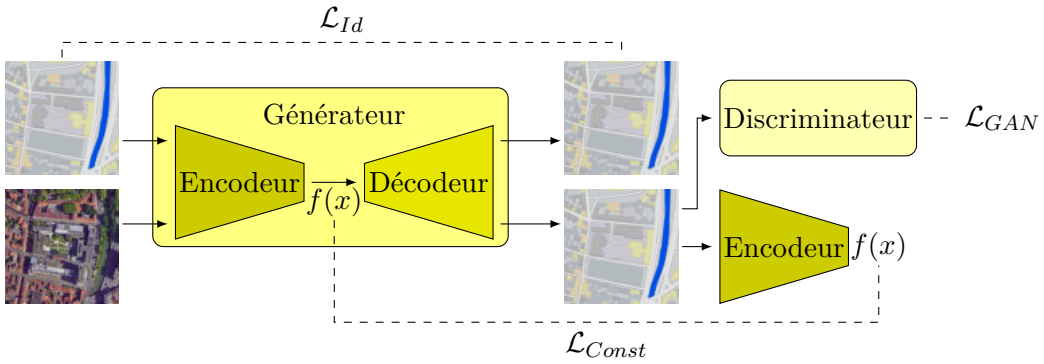


FIGURE 6.11 – Fonctionnement du *Domain Transfer Network* [Taigman et al., 2016] . Le générateur prend en entrée une image, qu'il transforme en image du domaine cible. L'image construite par le générateur est analysée par le discriminateur qui détermine si l'image est réelle ou générée, et calcule la fonction de perte L_{GAN}. Deux fonctions de régularisation interviennent pour l'entraînement du générateur : une fonction d'invariabilité des caractéristiques extraites par son encodeur L_{Const}, et une fonction d'invariabilité de transformation d'une image du domaine cible L_{Id}.

avec un terme d'auto-régularisation qui est calculé en effectuant la différence entre l'image source et l'image générée. Le fonctionnement du modèle est illustré en Figure 6.10. Ainsi, l'apprentissage du générateur est guidé par la somme de la fonction de perte du GAN et de la différence entre l'image donnée en entrée du générateur et l'image obtenue en sortie. Si cette approche fonctionne dans le cas présenté par [Shrivastava et al., 2017], où la différence entre l'image source et l'image obtenue est relativement faible, il semble qu'une différence directe entre l'image source et l'image générée ne peut pas fonctionner lorsque les deux images sont dans des domaines trop différents.

Une méthode présentée par [Chen et al., 2018a] utilise la traduction d'image à image de données non appariées pour la transformation de photos en dessins. Ici, la fonction de régularisation utilisée est une fonction qui cherche à préserver le contenu. Elle est calculée à l'aide du réseau de neurones VGG [Simonyan and Zisserman, 2015], de la même façon que [Gatys et al., 2016] pour le transfert de style.

Dans une autre approche, les auteurs de [Taigman et al., 2016] proposent le *Domain Transfer Network*. En plus du modèle de réseaux antagonistes génératifs (GAN), on ajoute une fonction f, qui prend en entrée des images du domaine source et des images générées, et dont la sortie doit être identique pour une image source et l'image générée associée. Ainsi, en plus de la fonction de perte associée au GAN, les auteurs ajoutent un paramètre de régularisation par la contrainte d'invariabilité de la fonction f. Dans la pratique, le générateur est un encodeur-décodeur, et la fonction f est représentée par la partie encodeur du réseau. Ainsi, lorsqu'on donne en entrée une image du domaine source ou son image associée construite par le générateur, les caractéristiques extraites par l'encodeur doivent

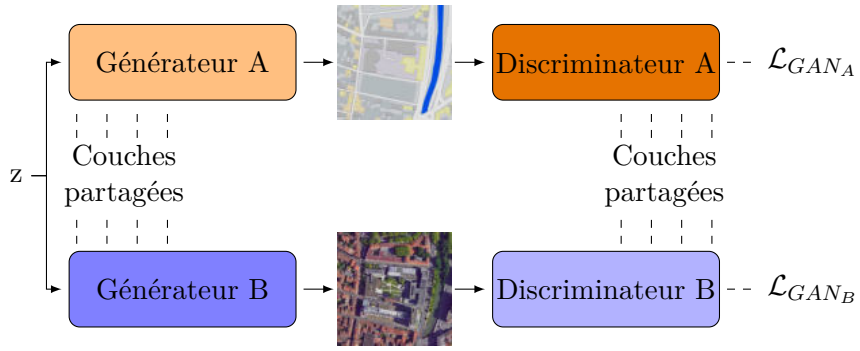


FIGURE 6.12 – Illustration du fonctionnement du CoGAN [Liu and Tuzel, 2016]. Ce modèle contient deux paires de GANs, pour générer des images dans deux domaines différents. Afin que les images générées partagent les mêmes informations sur le contenu, les premières couches des générateurs partagent les mêmes poids, comme les dernières couches des discriminateurs. On observe que ce n'est pas un modèle de traduction d'image à image, mais plutôt un modèle de génération conjointe d'images de domaines différents.

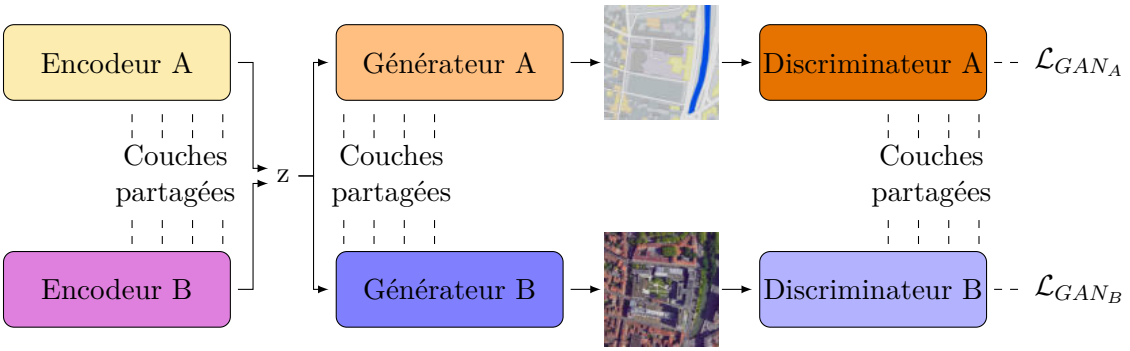


FIGURE 6.13 – Illustration du fonctionnement du modèle UNIT [Liu et al., 2017]. Ce modèle est un CoGAN auquel on a ajouté, en amont des générateurs, deux encodeurs qui partagent leurs dernières couches. Ceci permet aux caractéristiques extraites des images d'évoluer dans le même espace latent, qui est ensuite utilisé par les générateurs pour reconstruire ou traduire les images.

être identiques. De plus, le générateur doit reproduire une image identique lorsqu'on lui fournit une image du domaine cible, ce qui ajoute une deuxième régularisation. Le principe de fonctionnement du modèle est résumé en Figure 6.11.

Certaines méthodes ont recours à l'utilisation de plusieurs GAN en parallèle. Le CoGAN pour *Coupled Generative Adversarial Networks*, est un modèle proposé par [Liu and Tuzel, 2016]. Contrairement aux autres approches, ce modèle est utilisé pour construire des paires d'images avec la même information haut niveau, comme la géométrie, mais une information de bas niveau, comme le style, différente. Le modèle apprend une distribution conjointe et multi-domaine à l'aide d'une paire de GANs, où chaque GAN se focalise sur un domaine. Comme on souhaite que les deux images générées partagent la même

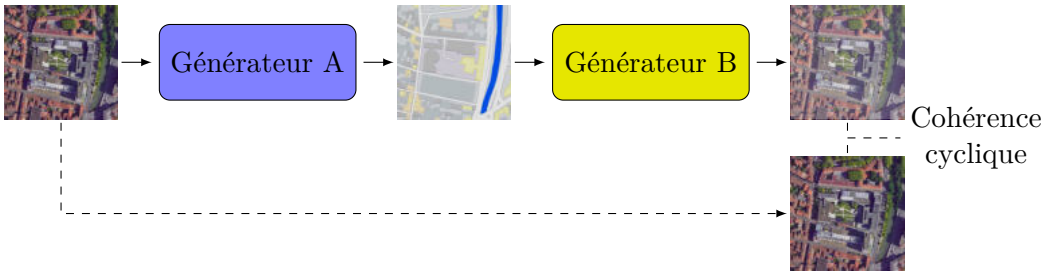


FIGURE 6.14 – Illustration du principe de la cohérence cyclique. Suite à une double transformation, l'image résultante doit être identique à l'image originale. La mesure de la différence entre ces deux images permet d'obtenir l'erreur de cohérence cyclique.

information de haut niveau, les poids des premières couches des générateurs sont partagés entre eux, tout comme ceux des dernières couches des discriminateurs sont partagés entre les discriminateurs. L'architecture du modèle est illustrée en Figure 6.12. Ce modèle n'est pas à proprement parler un modèle de traduction d'image à image, mais plutôt un modèle de génération conjointe d'images de domaines différents. Une extension du CoGAN est le modèle UNIT [Liu et al., 2017]. Ce modèle reprend le CoGAN pour le transformer en un outil de traduction d'image à image. En amont des générateurs, les auteurs ajoutent deux encodeurs qui partagent leurs dernières couches. Ainsi, ces auto-encodeurs partagent le même espace latent de contenu, c'est-à-dire que les caractéristiques extraites des images des deux domaines évoluent dans le même espace qui contient l'information sur le contenu des images. Les générateurs prennent alors en entrée des caractéristiques dans cet espace latent, pour reconstruire des images ou les traduire. Le fonctionnement de ce modèle est présenté en Figure 6.13. Une version pour l'"apprentissage en quelques coups" (*few shot learning*) a été proposée par [Liu et al., 2019].

Des approches généralistes, basées sur un double apprentissage ont ensuite été proposées par [Yi et al., 2017] avec le DualGAN, [Kim et al., 2017] avec le DiscoGAN et [Zhu et al., 2017] avec le CycleGAN. Dans ces travaux, il n'y a plus de domaine source et de domaine cible : on cherche à effectuer les traductions dans les deux sens. Ainsi, on peut ajouter une nouvelle forme de régularisation. Ces trois travaux utilisent le même principe de **cohérence cyclique**, c'est-à-dire qu'une double transformation domaine A vers domaine B vers domaine A doit donner une image identique à l'image de départ (et de même pour une transformation de B vers A vers B)[1]. La cohérence cyclique est illustrée en Figure 6.14. Le fonctionnement des trois modèles est présenté en Figure 6.15. Les trois modèles ont des architectures différentes pour leurs générateurs et discriminateurs, mais les deux générateurs de chaque modèle, comme les deux discriminateurs, ont la même architecture (mais des poids différents) :

1. On peut par ailleurs vérifier que le modèle UNIT satisfait aussi la contrainte de cohérence cyclique, grâce à l'espace latent partagé par les deux encodeurs

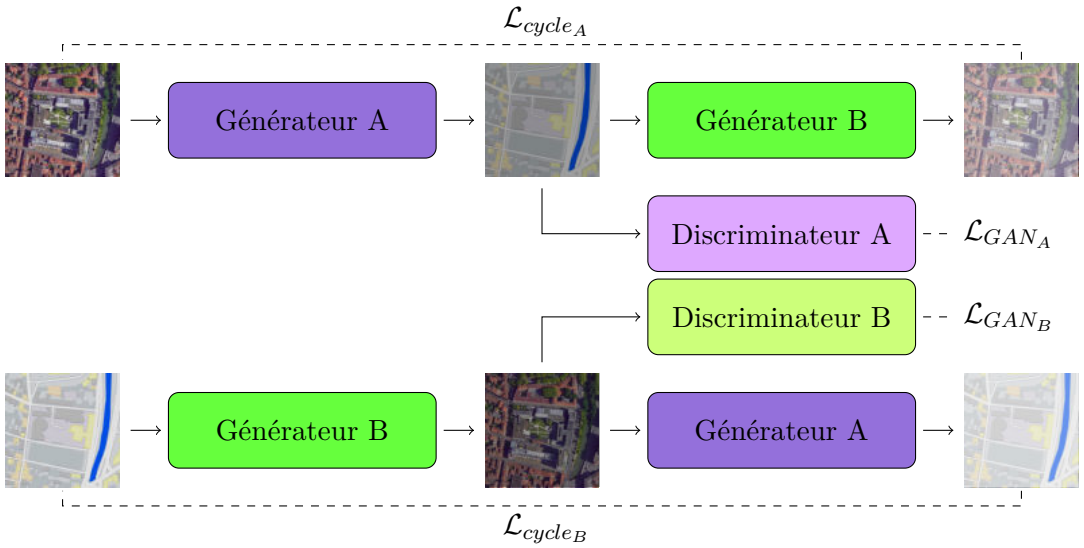


FIGURE 6.15 – Fonctionnement du CycleGAN [Zhu et al., 2017], du DualGAN [Yi et al., 2017] et du DiscoGAN [Kim et al., 2017]. Les trois modèles fonctionnent de la même façon : ils combinent deux sous-modèles de GAN (ici, le modèle A, en violet et le modèle B, en vert), où chacun est entraîné pour faire la traduction d'image à image d'un domaine vers un autre. On a donc deux fonctions de perte liées aux GANs ($\mathcal{L}_{GAN_A}$ et $\mathcal{L}_{GAN_B}$), plus deux mesures de la cohérence cyclique $\mathcal{L}_{cycle_A}$ et $\mathcal{L}_{cycle_B}$, qui évaluent l'écart entre une image source et l'image obtenue lorsqu'on applique deux transformations successives, pour chacun des deux domaines. La différence entre les trois modèles se fait au niveau de l'utilisation des mesures de cohérence cyclique : le DiscoGAN va traiter les deux mesures séparément sous la forme de deux pertes de reconstruction (une par domaine, donc une par générateur), alors que le CycleGAN et le DualGAN les voient comme une unique perte de cohérence cyclique, appliquée de la même manière aux deux générateurs.

— pour le DiscoGAN, on a des générateurs de type encodeur-décodeur (voir Section 3.4.2), et des discriminateurs classiques : ce sont des réseaux convolutifs qui proposent une unique valeur en sortie ;
— le CycleGAN utilise des générateurs structurés autour de blocs résiduels (voir Section 3.4.2), et les discriminateurs sont des PatchGANs ;
— les générateurs du DualGAN sont des U-net, et les discriminateurs sont des PatchGANs.

Ainsi, au vu de la proximité entre les différentes approches et de leur résultats, il semble que l'architecture des réseaux de neurones soit adaptable au problème particulier à résoudre.

Enfin certaines approches multimodales telles que le StarGAN [Choi et al., 2018], permettent de faire de la traduction d'image à image multi-domaine : dans ce cas là, on cherche à conditionner la génération d'images à l'aide d'un indicateur de domaine attendu. Par exemple, à partir d'une image source contenant un visage, on peut demander une tranfor-

mation par rapport à l'âge, la couleur des cheveux, l'émotion exprimée, etc. Le StarGAN utilise un unique GAN, où le générateur est capable d'apprendre plusieurs correspondance entre domaines. Le générateur est entraîné de manière cyclique, c'est-à-dire que dans un premier temps, il traduit une image originale dans le domaine cible demandé par son conditionnement, et cette image traduite est dans un second temps transformée dans le domaine de l'image originale. Le discriminateur est entraîné de manière à reconnaître les vraies images des fausses et à fournir une classification de l'image selon son domaine d'appartenance. De nombreux travaux ont leur version multimodale : on trouve par exemple le Multimodal UNIT (MUNIT) [Huang et al., 2018] ou le CycleGAN augmenté [Almahairi et al., 2018].

Dans notre cas, il n'est pas possible d'utiliser une approche basée sur des ensembles de données appariés. En effet, un tel algorithme nous demanderait d'associer à chaque image de profondeur de synthèse une image de profondeur réelle. Ceci nécessiterait donc d'annoter nos images de synthèse avec une image réelle où l'opérateur est dans la même posture. Cette tâche étant difficile et très coûteuse, elle nous paraît impossible à mettre en œuvre, d'autant plus que ce procédé devrait être répété à chaque installation d'un dispositif, car les opérateurs ont des morphologies différentes et ne portent pas les mêmes vêtements réfléchissants. Il ne nous semble pas non plus envisageable d'avoir recours à un procédé de transfert de style : celui-ci demandant de multiples itérations afin de transformer complètement une image, il serait trop long à mettre en place dans notre cas où nous souhaitons transformer une grande quantité de données.

Ainsi, nous avons décidé d'utiliser des algorithmes de traduction d'image à image avec des données non appariées. Plus en particulier, nous nous sommes focalisés sur le modèle du CycleGAN plutôt que sur le modèle de CoGAN, car les CycleGANs permettent de traduire directement une image d'un premier domaine en une image d'un autre domaine, au contraire du CoGAN. Nous présentons le modèle de CycleGAN utilisé dans la section suivante.

6.3 Traduction d'image à image

Le modèle CycleGAN [Zhu et al., 2017] est une approche non supervisée pour la traduction d'image à image qui ne nécessite pas de données appariées. Le CycleGAN permet de trouver une correspondance entre les distributions des deux ensembles de données. Dans ce qui suit, nous présentons l'architecture décrite par la Figure 6.16 avec les détails de l'entraînement et fournissons les premiers résultats sur le jeu de données de synthèse.

Dans la suite, nous manipulons plusieurs types de données, qui sont identifiées comme suit :

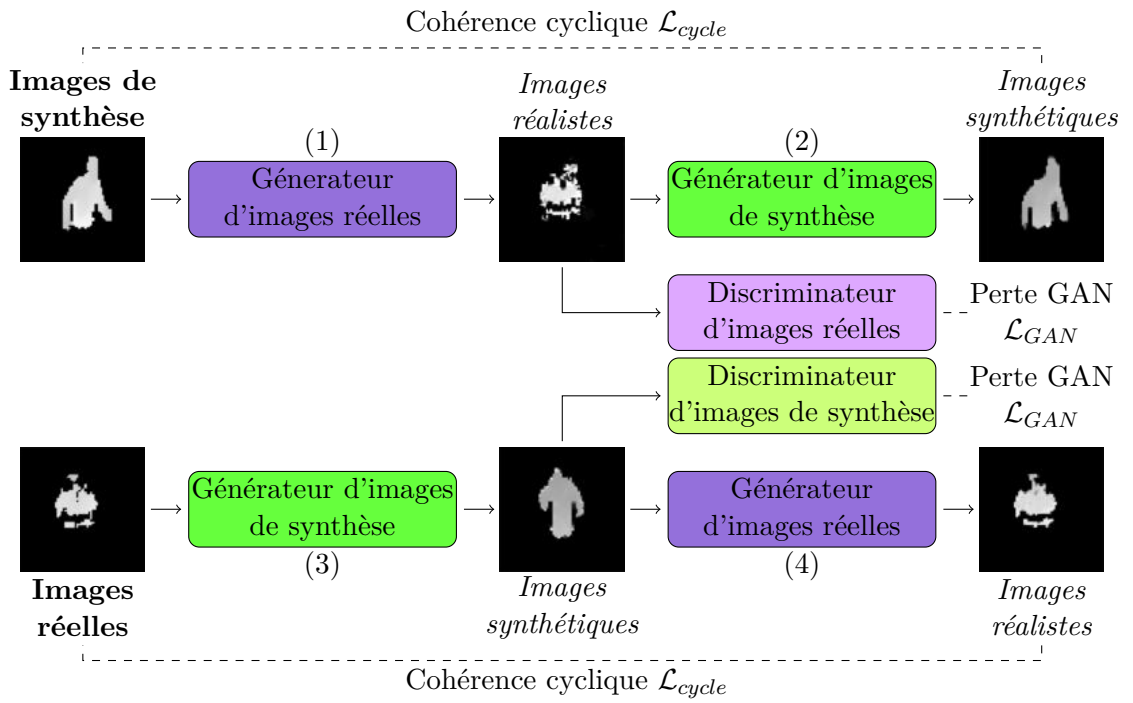


FIGURE 6.16 – Vue d'ensemble de notre procédé de traduction d'image à image. Ce modèle d'apprentissage profond s'articule autour de quatre réseaux : deux générateurs et deux discriminateurs. Le générateur d'images réelles (en violet) prend en entrée des **images de synthèse** (1) ou des *images synthétiques* (4), qu'il transforme en *images réalistes*. En parallèle, le générateur d'images de synthèse (en vert) prend en entrée des *images réelles* (3) ou des *images réalistes* (2), qu'il transforme en images synthétiques. Deux discriminateurs cherchent à différencier les **images réelles** des *images réalistes* (violet clair) et les **images de synthèse** des *images synthétiques* (vert clair).

— les **images de profondeur de synthèse** : ce sont les images qui ont été créées à l'aide des logiciels MakeHuman et Blender, et dont le processus de construction a été donné en Section 5.1. Elles sont parfois aussi appelées **images de synthèse** ;
— les **images de profondeur réelles** : ce sont des images qui ont été acquises dans un centre de tri. Elles sont parfois aussi appelées **images réelles** ;
— les *images de profondeur synthétiques* : elles représentent la transformation des **images de profondeur réelles** en images semblables à des **images de profondeur de synthèse**. Elles sont parfois aussi appelées *images synthétiques* ;
— les *images de profondeur réalistes* : elles représentent la transformation des **images de synthèse** en images semblables à des **images réelles**. Elles sont parfois aussi appelées *images réalistes*.

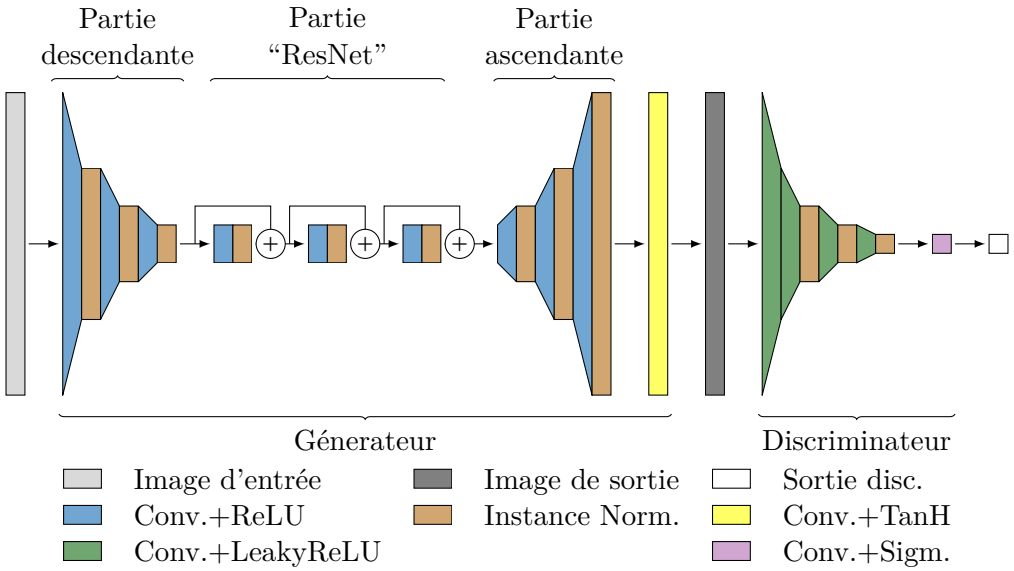


FIGURE 6.17 – Les architectures des générateurs et des discriminateurs de notre CycleGAN proposé. Les générateurs sont composés de trois parties : une partie descendante composée de couches convolutives, une partie "ResNet" composée de blocs résiduels et une partie ascendante composée de couches déconvolutives. Les discriminateurs sont des PatchGANs [Isola et al., 2017, Shrivastava et al., 2017] construits en utilisant des couches convolutives avec *stride*.

6.3.1 Architecture du modèle

Le CycleGAN est constitué de deux réseaux générateurs et de deux discriminateurs. Comme nous avons besoin d'architectures légères pour obtenir des performances en temps réel dans notre cas d'utilisation industriel, et comme l'opérateur est situé au centre de l'image et n'occupe qu'une petite partie de l'image de profondeur, nous recadrons les images de profondeur pour ne conserver que l'opérateur et nous redimensionnons les images en 64×64×1.

Les architectures des différents éléments du modèle sont inspirées des architectures proposées par [Zhu et al., 2017] pour les générateurs et les discriminateurs, et sont illustrées en Figure 6.17.

Instance Normalization

[Ulyanov et al., 2016] a montré que les modèles de réseaux de neurones ayant pour objectif de réaliser un transfert de style sont plus efficaces lorsqu'on remplace les couches de *Batch Normalization* par des couches d'*Instance Normalization*.

Soit $x \in \mathbb{R}^{T \times C \times L \times H}$ un tenseur contenant un *batch* de T éléments de C caractéristiques, d'une longueur L et d'une hauteur H. On note x_{tijk} sa $tijk$-ème valeur, avec j et k ses dimensions spatiales, i l'index de la caractéristique et t l'index de l'élement dans le *batch*.

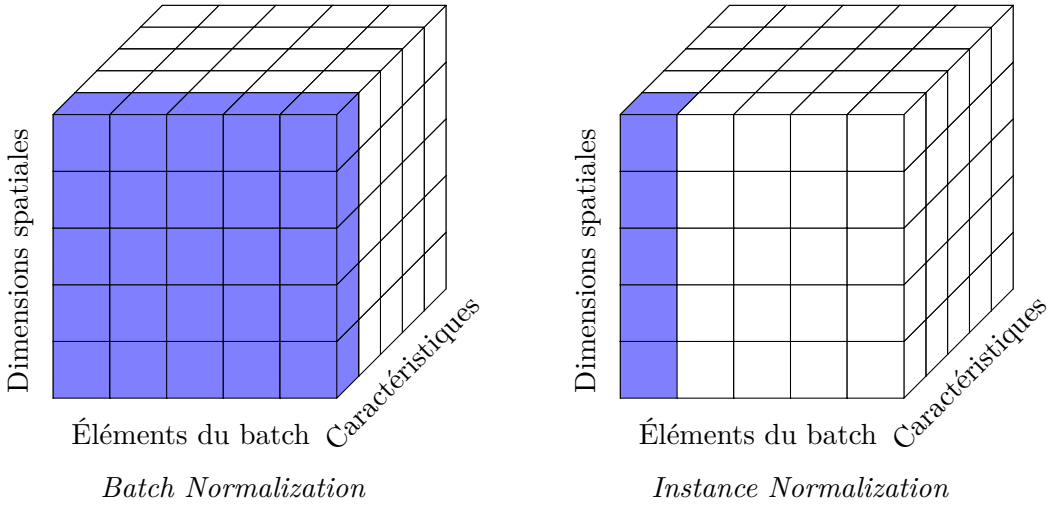


FIGURE 6.18 – La différence entre la *Batch Normalization* (à gauche) et l'*Instance Normalization* (à droite). Un exemple de valeurs normalisées ensemble sont indiquées en bleu. La *Batch Normalization* normalise ensemble tous les éléments du *batch*, caractéristique par caractéristique. En revanche, l'*Instance normalisation* normalise de manière indépendante chaque élément du batch, une nouvelle fois caractéristique par caractéristique.

La *Batch Normalization* est définie de la manière suivante [Ioffe and Szegedy, 2015] :

$$y_{tijk} = \frac{x_{tijk} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}}, \ \mu_i = \frac{1}{HWT}\sum_{t=1}^{T}\sum_{j=1}^{W}\sum_{k=1}^{H} x_{tijk}, \ \sigma_i^2 = \frac{1}{HWT}\sum_{t=1}^{T}\sum_{j=1}^{W}\sum_{k=1}^{H}(x_{tijk} - \mu_i). \quad (6.5)$$

L'*Instance Normalization* est définie de la manière suivante [Ulyanov et al., 2016] :

$$y_{tijk} = \frac{x_{tijk} - \mu_i ti}{\sqrt{\sigma_{ti}^2 + \epsilon}}, \ \mu_{ti} = \frac{1}{HW}\sum_{j=1}^{W}\sum_{k=1}^{H} x_{tijk}, \ \sigma_{ti}^2 = \frac{1}{HW}\sum_{j=1}^{W}\sum_{k=1}^{H}(x_{tijk} - \mu_{ti}). \quad (6.6)$$

Ainsi, la *Batch Normalization* normalise ensemble tous les éléments d'un *batch*, alors que l'*Instance Normalization* normalise chacun des éléments séparément, ce qui simplifie l'apprentissage [Ulyanov et al., 2016].

Architecture des générateurs

Pour déterminer l'architecture des générateurs du modèle de CycleGAN que nous utilisons, nous avons effectué plusieurs tests, dont un tableau récapitulatif est donné en Tableau 6.1. Nous avons directement essayé plusieurs types d'architectures différentes, afin de sélectionner celle qui donnait les meilleurs résultats visuels, à défaut de pouvoir effectuer une mesure de qualité.

D'après ces tests, l'architecture finale des deux générateurs est la même et se compose

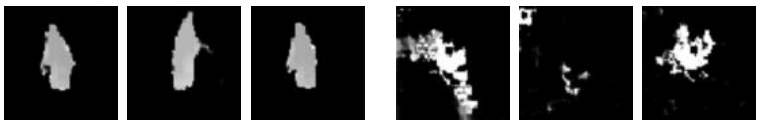
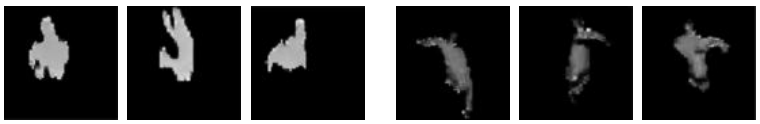
	Images réelles	**Image synthétiques**
Images sources		
Type d'architecture	*Images synthétiques*	*Images réalistes*
Séquentiel		
Encodeur-Décodeur		
U-net		
ResNet		

TABLE 6.1 – Tableau récapitulatif des principales architectures de générateurs utilisées pour le module de traduction d'image à image par CycleGAN. On remarque que certains types d'architecture ne sont pas en mesure de générer des images pertinentes, en particulier au niveau des *images synthétiques* de l'architecture séquentielle et du U-net. Les *images réalistes* sont plus convaincantes, même si l'architecture séquentielle fonctionne mal et que l'Encodeur-Décodeur a un problème de normalisation.

de trois parties :

— une première partie descendante composée de trois couches convolutives avec un décalage pour réduire la dimension spatiale des images, le nombre de filtres doublant à chaque couche convolutive, allant ainsi de 32 jusqu'à 128 ;

— une deuxième partie contenant trois blocs résiduels [He et al., 2016] d'une seule couche convolutive 2D de 128 filtres ;

— une troisième partie ascendante constituée de trois couches déconvolutives avec un décalage pour retrouver la taille originale de l'image, avec le nombre de filtres diminuant progressivement de 128 à 32.

Toutes les couches convolutives et déconvolutives ont une taille de noyau de 4×4, avec du remplissage par zéro (en anglais *zero padding*), et sont suivies d'une fonction d'activation ReLU et d'une couche d'*Instance Normalization* [Ulyanov et al., 2016]. La couche finale est une couche convolutive 2D avec un seul filtre, une taille de noyau de 4×4 et une activation utilisant la fonction tangente hyperbolique pour obtenir notre image traduite

normalisée [Radford et al., 2015].

Architecture des discriminateurs

Les discriminateurs sont deux 64×64 PatchGANs [Isola et al., 2017, Shrivastava et al., 2017], qui sont des réseaux de neurones entièrement convolutifs dont le but est de classer si des patchs superposés extraits de l'image sont authentiques ou construits par un réseau de neurones, en produisant une carte de probabilité. Plus précisément, les deux discriminateurs contiennent quatre couches convolutives avec décalage, un nombre croissant de filtres de 32 à 256, une taille de noyau de 4×4, avec remplissage par zéro, et sont suivis d'une activation LeakyReLU avec $\alpha = 0,2$. Elles sont toutes suivies d'une couche d'*Instance Normalization*, à l'exception de la première qui n'est pas normalisée. Une dernière couche convolutive avec un seul filtre et une fonction d'activation sigmoïde produit la discrimination des patchs.

6.3.2 Détails d'entraînement

Pour entraîner ce modèle, nous utilisons, comme décrit dans [Zhu et al., 2017], trois fonctions de perte : deux pertes adverses $\mathcal{L}_{GAN}$ et une perte de cohérence cyclique.

Pour deux générateurs $G_R : S \to R$, $G_S : R \to S$ où R et S sont respectivement les ensembles de données réelles et de synthèse, et leurs discriminateurs associés D_R et D_S, on exprime la fonction de perte adverse comme :

$$\mathcal{L}_{GAN}(G_R, D_R, S, R) = \mathbb{E}_{r \in R}[||D_R(r)||_2] + \mathbb{E}_{s \in S}[||1 - D_R(G_R(s))||_2] \tag{6.7}$$

où G_R a pour but de construire des *images réalistes* similaires aux **images réelles** de R et D_R a pour objectif de séparer les **échantillons réels** r des *échantillons réalistes générés*, et :

$$\mathcal{L}_{GAN}(G_S, D_S, R, S) = \mathbb{E}_{s \in S}[||D_S(s)||_2] + \mathbb{E}_{r \in R}[||1 - D_S(G_S(r))||_2] \tag{6.8}$$

où G_S a pour but de construire des *images synthétiques* similaires aux **images de synthèse** de S et D_S a pour objectif de séparer les **échantillons de synthèse** s des *échantillons synthétiques générés*.

Nous utilisons une fonction de perte L_2 plutôt qu'une log-vraisemblance négative car cette perte est plus stable pendant l'entraînement et génère des résultats de meilleure qualité [Mao et al., 2016].

En parallèle, l'objectif de la fonction de perte de cohérence cyclique est de s'assurer que $G_S(G_R(s)) \approx s$, c'est-à-dire qu'un cycle de traduction ramène s à l'image de synthèse originale. Il en va de même pour les images réelles, où nous voulons nous assurer que

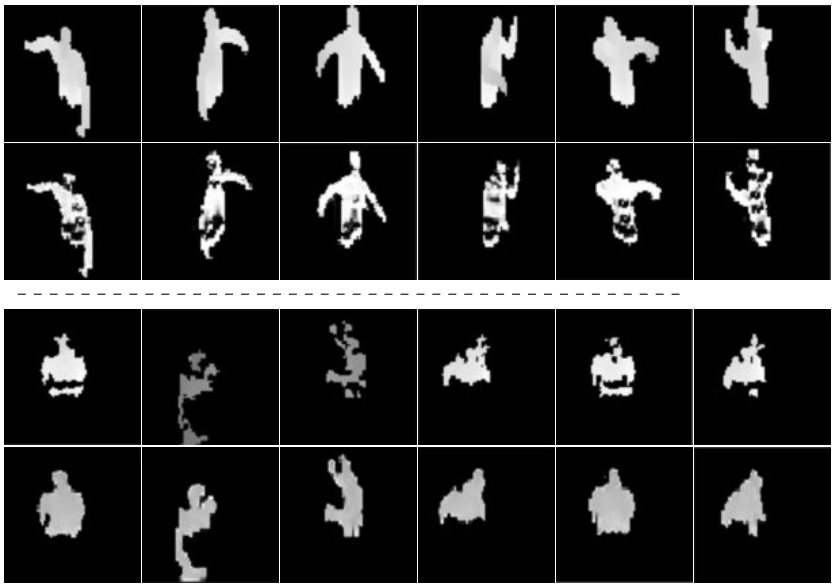


FIGURE 6.19 – Exemple de traductions d'image à image réalisées par notre CycleGAN. Nous pouvons voir que la posture de l'utilisateur est préservée lors des deux traductions. Première ligne : **images de synthèse** de notre ensemble de **données de synthèse**. Deuxième ligne : *images réalistes* traduites associées aux images de la première ligne. Troisième ligne : **images réelles** de notre ensemble de **données réelles**. Dernière ligne : *images synthétiques* traduites associées aux **images réelles** de la troisième ligne.

$G_R(G_S(r)) \approx r$. Cette perte de cohérence de cycle est alors exprimée comme suit :

$$\mathcal{L}_{cycle}(G_S, G_R) = \mathbb{E}_{s\in S}[||G_S(G_R(s)) - s||_1] + \mathbb{E}_{r\in R}[||G_R(G_S(r)) - r||_1] \tag{6.9}$$

Finalement, la perte globale pour entraîner notre modèle est :

$$\begin{aligned}\mathcal{L}(G_S, G_R, D_S, D_R) = &\mathcal{L}_{GAN}(G_R, D_R, S, R) \\ &+ \mathcal{L}_{GAN}(G_S, D_S, R, S) \\ &+ \lambda\mathcal{L}_{cycle}(G_S, G_R)\end{aligned} \tag{6.10}$$

où λ contrôle l'importance relative entre les deux types de perte.

Nous avons essayé plusieurs valeurs différentes de λ avant de le fixer à 10, comme conseillé dans l'article original introduisant le modèle CycleGAN [Zhu et al., 2017]. Nous entraînons les réseaux à l'aide de l'optimiseur Adam [Kingma and Ba, 2015] avec un taux d'apprentissage de 10^{-6}.

FIGURE 6.20 – Exemple de séquences d'images traduites par notre module de traduction d'image à image. Nous pouvons voir que la traduction semble consistente tout au long de la séquence. Première ligne : **images de synthèse** de notre ensemble de **données de synthèse**. Deuxième ligne : *images réalistes* traduites associées aux images de la première ligne. Troisième ligne : **images réelles** de notre ensemble de **données réelles**. Dernière ligne : *images synthétiques* traduites associées aux **images réelles** de la troisième ligne.

6.3.3 Résultats préliminaires

Certains résultats sont présentés en Figures 6.19 et 6.20. Nous voyons que le générateur d'images réelles parvient à dégrader de manière réaliste les images de synthèse en ajoutant du bruit sur le corps de l'opérateur humain et en enlevant des parties, comme dans une image réelle. D'autre part, le générateur d'images de synthèse semble remplir des trous et assembler des parties d'images réelles pour les rendre plus "synthétiques". Nous observons de plus que les traductions restent cohérentes tout au long d'une séquence. Toutefois, et comme prévu, les deux générateurs n'ajoutent ni n'enlèvent des objets autour de l'utilisateur. On remarque que les **images de synthèse** semblent plus faciles à traduire en *images réalistes* que les **images réelles** en *images synthétiques*.

Les *images de profondeur réalistes* étant devenues visuellement plus proches des **images réelles**, nous espérons qu'un réseau de neurones entraîné sur celles-ci sera en mesure de fournir une estimation de posture 3D cohérente sur les **images de profondeur réelles**. C'est l'objet de la partie suivante.

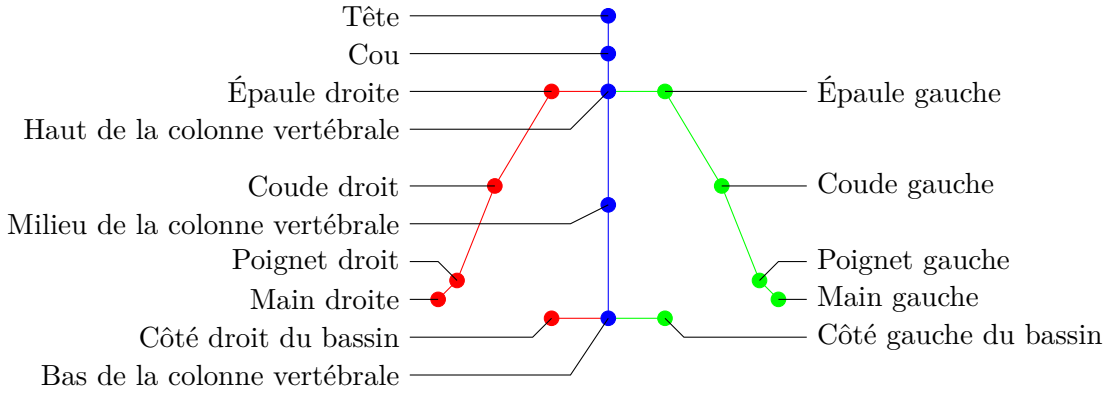


FIGURE 6.21 – Les quinze articulations utilisées pour définir la posture 3D du haut du corps.

6.4 Estimation de posture sur les images réelles

Nous effectuons une estimation de la posture 3D sur des images de profondeur en utilisant un réseau de neurones convolutif. Comme nos **jeux de données réels** ne contiennent pas d'annotations de posture humaine en 3D, nous ne pouvons pas entraîner directement notre réseau sur ces données. Cependant, grâce à notre ensemble de **données de synthèse** et au CycleGAN entraîné précédemment, nous pouvons générer des *images de profondeur réalistes* et annotées. Ce processus peut idéalement produire un ensemble illimité d'*images de profondeur réalistes* couplées à l'annotation de posture humaine 3D correspondant aux **images de synthèse** données en entrée. Nous utilisons ces couples entre des *images de profondeur réalistes* et les annotations de posture humaine en 3D des **images de synthèse** pour entraîner notre réseau d'estimation de posture.

Comme précédemment (voir Chapitre 5), la posture 3D est définie par les quinze articulations du haut du corps suivantes : *Bas de la colonne vertébrale*, *Milieu de la colonne vertébrale*, *Haut de la colonne vertébrale*, *Cou*, *Tête*, *Épaule gauche*, *Coude gauche*, *Poignet gauche*, *Main gauche*, *Épaule droite*, *Coude droit*, *Poignet droit*, *Main droite*, *Côté gauche du bassin* et *Côté droit du bassin*. Ces articulations sont illustrées en Figure 6.21.

Le réseau d'estimation de la posture 3D est un réseau de neurones convolutif directement inspiré des travaux présentés en Chapitre 5. Ses entrées sont des images de profondeur de taille 64×64, et il produit en sortie un vecteur à 45 dimensions représentant la position 3D des quinze articulations du corps. Il contient trois blocs de trois couches convolutives 2D de 64 filtres de taille 4×4, avec une fonction d'activation ReLU et une couche de *batch normalization*. Les blocs sont séparés par des couches de *max-pooling*. La couche finale est une couche dense de 15×3 neurones, pour estimer la posture 3D des quinze articulations, c'est-à-dire $15 \times 3 = 45$ coordonnées. Une illustration de cette architecture est donnée en

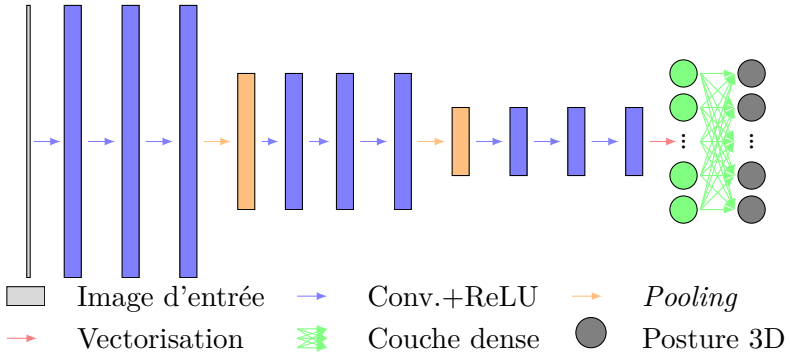


FIGURE 6.22 – L'architecture de notre réseau de neurones d'estimation de posture 3D. Le réseau est composé de trois blocs contenant trois couches convolutives 2D de 64 filtres 4×4 utilisant une activation ReLU, séparés par des couches de *max-pooling*. Une dernière couche dense effectue la régression des caractéristiques extraites de l'image vers une posture 3D composée de 45 valeurs représentant les quinze articulations qui définissent la posture.

Figure 6.22.

Pour entraîner ce modèle, la fonction de perte utilisée est l'erreur quadratique moyenne. Nous nous servons de l'optimiseur Adam [Kingma and Ba, 2015], avec un taux d'apprentissage de 10^{-3} et des lots de 32 images.

Des exemples de résultats d'estimation de la posture sur des images réelles sont présentés en Figure 6.23, puis dans la Figure 6.24 sur une séquence d'images. Nous pouvons voir que nos résultats semblent cohérents avec les images et restent consistants dans le temps, mais la qualité de l'estimation de posture dépend fortement de la variabilité des images de profondeur de synthèse utilisées pour l'apprentissage et de la qualité de la segmentation des utilisateurs dans les images de profondeur. En effet, notre ensemble de données de synthèse manque par exemple d'images de synthèse avec les bras levés et, comme le montre la troisième ligne de la Figure 6.23, les postures estimées présentent souvent ce type d'erreur sur des images mal segmentées.

6.5 Évaluation de l'approche

Nous comparons les différents résultats grâce à plusieurs mesures [Zhang et al., 2019] telles que le pourcentage de points-clés corrects (en anglais *Percentage of Correct Key-Points*, abrégé PCK) à 150mm (PCK@150mm) et 80mm (PCK@80mm), qui représente le pourcentage de points-clés dont la distance entre la position estimée par l'algorithme et la position réelle est inférieure à un certain seuil (ici 150mm et 80mm), l'erreur moyenne par articulation (en anglais *Mean Per Joint Position Error*, abrégé MPJPE), qui mesure la distance euclidienne moyenne entre la position estimée des points-clés et leur position attendue, et l'erreur moyenne par articulation après analyse de Procrustes (en anglais

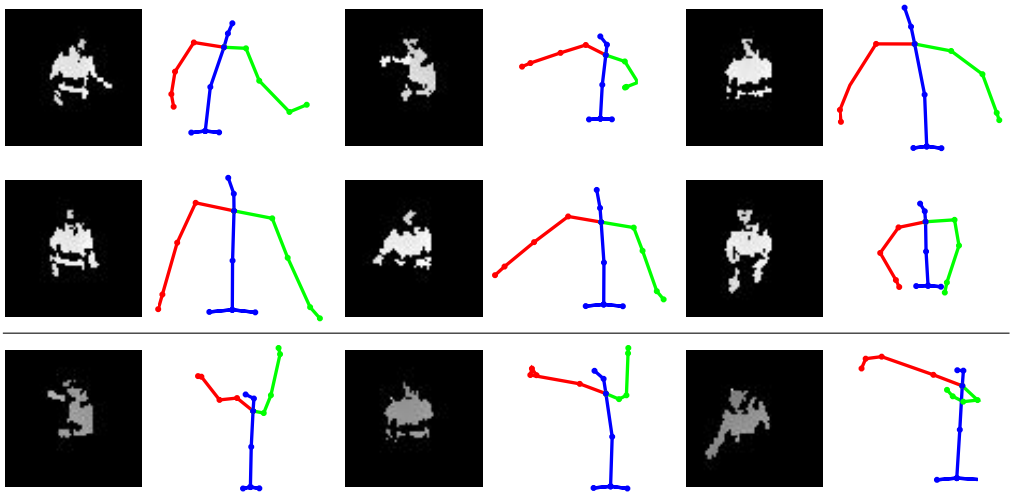


FIGURE 6.23 – Exemples d'images de profondeur réelles et leur estimation de posture 3D associée. Deux premières lignes : les postures semblent cohérentes avec l'image de profondeur de l'utilisateur. Troisième ligne : exemple d'échecs de notre approche. Les postures mal estimées affichent souvent des bras en l'air. Après analyse, ces erreurs se produisent souvent lorsqu'il y a des défauts dans la segmentation, par exemple des pixels de l'arrière plan brisant la normalisation de la profondeur de l'image (que nous pouvons voir ici avec une nuance de gris différente sur les images de la troisième ligne).

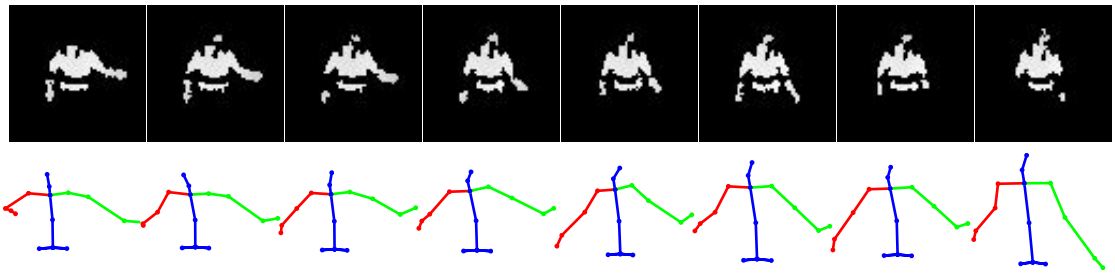


FIGURE 6.24 – Prédictions de l'estimateur de posture sur une séquence d'images réelles. On observe que les postures estimées restent cohérentes tout au long de la séquence. On voit cependant de légères oscillations, dues au fait que chaque posture est estimée indépendamment des postures précédentes et suivantes, et qu'il n'y a pas de lissage.

Type de données	Type d'entraînement	PCK 150mm	PCK 80 mm	MPJPE 3D	PAMPJPE 3D
Normales	Supervisé	100.0	100.0	10.9	9.9
Bruitées	Supervisé	100.0	100.0	21.4	18.8
&	CycleGAN	**93.3**	**46.0**	**84.3**	**65.9**
Degradées	Non supervisé	55.0	0.2	148.0	87.6

TABLE 6.2 – Résultats de notre réseau d'estimation de posture dans diverses conditions d'entraînement, pour plusieurs types de données de synthèse. Nous observons que l'utilisation d'un CycleGAN améliore les résultats par rapport au cadre entièrement non supervisé, bien qu'ils soient toujours moins bons que ceux obtenus par un apprentissage totalement supervisé.

Procrustes Analysis Mean Per Joint Position Error, abrégé PAMPJPE), qui est une erreur moyenne par articulation où une transformation de similitude est appliquée avant de calculer la distance euclidienne. Pour les mesures PCK, on cherche à avoir la valeur la plus grande possible (au maximum 100%), alors que pour les MPJPE et PAMPJPE, données en millimètres, on souhaite obtenir l'erreur la plus faible possible (au minimum 0mm).

Première étude

Afin de pouvoir mesurer la qualité de nos résultats, et comme nous ne disposons pas d'annotation sur les données réelles, nous avons créé un nouvel ensemble de données dégradées en appliquant une combinaison de bruit blanc et de suppression de parties pour construire l'ensemble de données bruitées et dégradées. Nous effectuons une étude d'ablation sur ce deuxième ensemble de données, contenant 200 000 images, ce qui nous permet de prouver que le bloc de traduction d'image à image que nous utilisons est intéressant pour améliorer la qualité de l'estimation de posture.

Comme le montre le Tableau 6.2, l'utilisation des sorties de notre réseau de traduction d'image à image améliore nettement la qualité des estimations de posture par rapport à l'entraînement direct du même réseau sur les images de profondeur de synthèse, même si les performances sont toujours inférieures à un entraînement totalement supervisé : le PCK@150mm et le PCK@80mm s'améliorent respectivement de 55.0% à 93,3% et de 0,2% à 46,0%, tandis que le MPJPE et le PAMPJPE passent de 148,0mm à 84,3mm (réduction de 43%) et de 87,6mm à 65,9mm (réduction de 25%).

Il montre que notre CycleGAN est capable de capturer correctement certaines caractéristiques des images de profondeur dégradées et de les traduire sur les images de synthèse. Il est également important de noter que la véritable difficulté de notre jeu de données réside dans la dégradation induite par le contexte industriel (c'est-à-dire les trous dans l'image en profondeur) et non dans l'imprécision du capteur (simulée par le bruit blanc).

Type d'entraînement	MPJPE 3D	PAMPJPE 3D
CycleGAN	**268.7**	**131.6**
Non supervisé	442.1	199.6

TABLE 6.3 – Résultats de notre estimation de posture par rapport à l'estimation de posture fournie par le capteur Kinect V2. En raison des faits exposés précédemment, il existe un écart important entre l'estimation de la posture calculée par le capteur Kinect et la nôtre. Cependant, l'utilisation du CycleGAN réduit considérablement cet écart et semble ainsi améliorer la qualité de l'estimation de la posture de notre réseau.

Deuxième étude

Afin de pouvoir effectuer davantage de mesures sur la qualité de nos résultats, nous avons acquis un deuxième ensemble de données réelles dans un cadre plus contrôlé, simulant un contexte industriel simplifié. Ces images ont été acquises avec un capteur Kinect V2 avec une vue plongeante sur l'utilisateur, nous permettant de comparer notre estimation de posture à celle directement fournie par le capteur. Cet ensemble de données contient environ 13 000 images en profondeur de plusieurs sujets dans une grande variété de postures. Cependant, la comparaison avec ces données est difficile pour plusieurs raisons :

— notre modèle de posture n'est pas le même que celui de la Kinect : les points d'intérêt similaires ne sont pas exactement situés au même endroit sur le corps humain ;
— nous estimons une posture humaine en 3D, alors que la Kinect V2 n'estime qu'une posture en 2D+Z : les deux premières coordonnées (X,Y) sont estimées en coordonnées pixelliques sur l'image de profondeur et ensuite traduites dans l'espace réel grâce aux paramètres intrinsèques de la caméra, mais la troisième est calculée en ajoutant un décalage donné à la valeur de profondeur du pixel, qui peut être très éloigné de la position réelle de l'articulation lorsque le capteur n'est pas devant l'utilisateur ;
— l'estimation de la posture de la Kinect V2 est calibrée pour une utilisation dans un contexte de divertissement, où le capteur se trouve devant l'utilisateur. Dans notre réglage, le capteur a une vue plongeante sur l'utilisateur, ce qui entraîne des déformations de la posture estimée.

Une illustration de ces problèmes est donnée en Figure 6.25. En particulier, nous pouvons voir que l'estimation de la posture proposée par la Kinect est fortement inclinée vers l'avant, alors que la nôtre reste droite, ce qui est plus proche de la position réelle de l'utilisateur.

Cependant, bien que nous ne puissions pas comparer nos résultats à une véritable vérité terrain, la comparaison confirme les résultats que nous avons montrés sur les données de synthèse bruitées et dégradées (voir Tableau 6.3). En raison des faits exposés précédemment, il existe un écart important entre l'estimation de posture proposée par le capteur Kinect V2 et la nôtre. Toutefois, l'utilisation du CycleGAN réduit considérablement cet

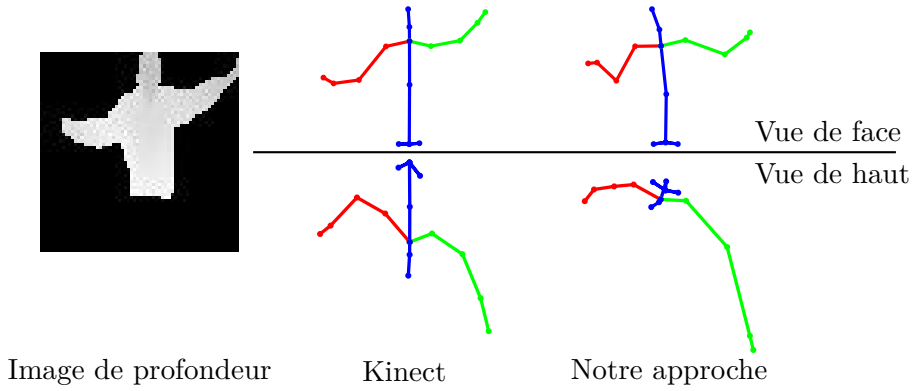


FIGURE 6.25 – Comparaison de l'estimation de la posture de la Kinect V2 avec l'estimation de posture proposée. À gauche : l'image de profondeur segmentée. Au centre : l'estimation de la posture 2D+Z fournie par le capteur Kinect V2. À droite : l'estimation de la posture proposée par notre approche. Nous pouvons voir que même si l'estimation de posture fournie par le capteur Kinect V2 et celle proposée par notre approche sont similaires lorsqu'elles sont vues de face, l'estimation de posture donnée par le capteur est sérieusement inclinée vers l'avant, ce qui rend difficile la comparaison précise de la qualité des deux estimations de posture.

écart et améliore ainsi la qualité de l'estimation de la posture du réseau : le CycleGAN permet d'améliorer l'estimation de posture en réduisant à la fois le MPJPE et le PAMPJPE, respectivement de 39% et 34%.

6.6 Conclusion

Nous proposons une méthode qui nous permet d'obtenir une estimation de posture 3D sur les images de profondeur réelles. Pour ce faire, nous utilisons un module de traduction d'image à image qui rend les images de synthèse plus réalistes. Ces données de profondeur réalistes, associées à l'annotation de posture des images de synthèse dont elles sont la transformation, servent ensuite à entraîner un modèle d'estimation de posture.

Cet estimateur de posture est alors capable de réaliser une estimation de posture 3D pertinente sur les images réelles.

En résumé

Dans ce chapitre, nous décrivons une contribution importante de cette thèse : la construction d'un processus d'apprentissage permettant de réaliser l'estimation de posture 3D sur des images de profondeur acquises en environnement industriel.

Dans un premier temps, nous présentons quelques concepts et travaux sur lesquels se sont appuyés les travaux proposés dans ce chapitre (voir Section 6.2). Premièrement, nous décrivons dans cette section les méthodes d'apprentissage qui peuvent être utilisées lorsqu'on possède à la fois des données annotées et des données non annotées. Deuxièmement, nous exposons les approches qui permettent de changer le style d'une image, dont le CycleGAN [Zhu et al., 2017], que nous utilisons pour effectuer notre traduction d'image à image.

Nous décrivons donc par la suite en Section 6.3 le procédé de raffinage d'images de profondeur de synthèse. L'objectif de ce module est de transformer des images de profondeur de synthèse, pour lesquelles on connaît la posture de l'opérateur, en images de profondeur réalistes, tout en conservant cette information posturale, afin que ces images transformées puissent être utilisées comme base de données pour l'entraînement d'un modèle dédié à l'estimation de posture 3D en aval.

Ensuite, nous nous servons de ces images traduites pour entraîner un modèle d'apprentissage profond pour l'estimation de posture 3D sur des images réelles. Les différents paramètres du modèle utilisé et son apprentissage sont présentés en Section 6.4.

Enfin, nous évaluons notre approche en Section 6.5. Nos expériences, appliquées aussi bien à des images de synthèse qu'à des images de profondeur réelles, acquises dans un contexte industriel simulé, nous permettent de prouver que l'utilisation de notre module de traduction d'image à image permet d'améliorer considérablement les performances du réseau de neurones qui estime la posture, par rapport à un réseau de neurones qui aurait directement été entraîné sur des images de synthèse.

Conclusion

Dans cette thèse, nous avons cherché à construire une information pertinente à partir de données imprécises ou incomplètes, avec une application industrielle à l'analyse d'activité d'opérateurs humains dans un centre de tri de déchets.

Pour ce faire, nous avons mis en place plusieurs processus qui sont résumés dans cette conclusion. Nous présentons ensuite les différentes perpectives qui s'ouvrent pour améliorer les systèmes développés pendant cette thèse.

1 Contributions

Pour résoudre le problème qui nous était posé, nous avons présenté dans cette thèse plusieurs contributions. Nous distinguons les contributions industrielles, intéressantes pour la société Ebhys, des contributions de recherche, intéressantes pour la communauté des chercheurs.

1.1 Contributions industrielles

Une première contribution industrielle analyse en détail le contexte et l'environnement au sein duquel le dispositif d'analyse d'activité est déployé (Chapitre 1). Nous avons introduit les différentes contraintes qu'impliquent l'environnement industriel, mais aussi celles imposées par l'entreprise et celles induites par la norme *NF X35—702*. Suite à cette analyse, nous avons pu choisir le type de capteur utilisé pour acquérir les données.

Une analyse ergonomique, menée conjointement avec des ergonomes, couplée à une étude approfondie de la norme *NF X35—702* nous a permis de définir des indicateurs pertinents pour la mesure de la pénibilité au travail d'un agent de tri. Ces indicateurs sont pour la plupart des indicateurs posturaux. Ces travaux ont fait l'objet d'une publication et d'une présentation au *Congrès de la Société d'Ergonomie de Langue Française* (SELF) en 2018 [Dutrieux et al., 2018].

Nous avons mis en place un premier prototype en conditions réelles. Ce prototype, installé en début d'année 2018, nous a permis d'acquérir les données de profondeur réelles

utilisées tout au long des travaux présentés dans cette thèse.

Nous avons ensuite pu établir un système de calcul des indicateurs ergonomiques à partir des images de profondeur brutes. Ce pipeline est composé de trois modules distincts : un module de segmentation d'images de profondeur, un module d'estimation de la posture 3D et un module de calcul des indicateurs ergonomiques à partir de l'estimation de posture. Ces trois modules ont été développés au cours de la thèse, et nous pouvons donc proposer un système complet et fonctionnel. Il est important de souligner que ce système, si il aurait pu s'apparenter à un système de vidéo-surveillance des opérateurs, est bien un dispositif ergonomique pour la santé des agents de tri.

1.2 Contributions de recherche

La première contribution de recherche est une méthode d'apprentissage profond faiblement supervisé. L'approche proposée s'appuie sur la mise à jour itérative d'un ensemble d'entraînement constitué de données imprécises, c'est-à-dire où les annotations ne peuvent pas être considérées comme des vérités terrains. Cette nouvelle méthode d'entraînement met itérativement à jour l'ensemble de données d'entraînement pour ne conserver que les données avec les annotations les plus pertinentes. Pour réaliser cette mise à jour, nous utilisons les prédictions du modèle réalisées au début de l'entraînement. Ceci permet de repérer les annotations aberrantes au cours de l'apprentissage et d'améliorer les résultats finaux du modèle. Ce travail a été publié à l'*IEEE International Conference on Image Processing* (ICIP) en 2019 [Blanc-Beyne et al., 2019].

Notre seconde contribution est une étude pour l'élaboration d'un modèle de réseau de neurones performant et léger pour l'estimation de posture 3D. Cette étude analyse les différents paramètres de l'architecture d'un réseau de neurones convolutif pour l'estimation de posture 3D sur des images de profondeur de synthèse. Ceci nous a permis de construire un modèle d'estimation de posture performant et léger sur des images de profondeur de synthèse, permettant par la suite de calculer les indicateurs ergonomiques.

Notre troisième contribution est un processus d'apprentissage permettant de réaliser l'estimation de posture 3D sur des images de profondeur non annotées. Pour ce faire, l'approche proposée se sert d'un ensemble de données d'images de profondeur de synthèse annotées. Ces images de synthèse sont transformées en image de profondeur réalistes par un module de traduction d'image à image. Ensuite, ces données de profondeur réalistes sont utilisées pour entraîner un réseau de neurones d'estimation de posture 3D. Ce modèle est ensuite capable de réaliser des estimations de postures fiables sur les images de profondeur réelles. Ce travail a été soumis et accepté à l'*European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (ECML-PKDD) en

2020.

2 Perspectives

Les travaux menés tout au long de cette thèse et la construction d'un dispositif d'analyse de l'activité d'opérateurs humains en environnement industriel ouvre naturellement de nombreuses perspectives. Une nouvelle fois, nous séparons les perspectives industrielles des perspectives de recherche.

2.1 Perspectives industrielles

Plusieurs perspectives industrielles sont apparues avec la construction du dispositif de mesure des indicateurs de pénibilité. On distingue ici plusieurs perspectives : une première perspective de test des résultats obtenus, une seconde perspective de déploiement du dispositif dans les centres de tri, et des perspectives d'amélioration du dispositif, par la mesure de nouveaux indicateurs ou par la généralisation de son fonctionnement.

Réalisation de tests en conditions réelles À court terme, une première perspective est la réalisation de tests, afin de vérifier les performances de notre dispositif en conditions réelles. En collaboration avec une équipe d'ergonomes, nous souhaitons effectuer une série de mesures afin de comparer les performances de notre système à un outil de mesure de la posture ayant recours à des marqueurs portés par les agents de tri. Ces mesures auraient dues être effectuées avant la fin de la thèse, mais l'épidémie de COVID-19 a retardé leur mise en place.

Mise en place dans des centres de tri Une perspective à moyen et long terme est le déploiement du système d'analyse d'activité des opérateurs humains dans des centres de tri. Le dispositif fait déjà partie d'un projet qui a gagné un appel d'offre pour la restructuration d'un centre de tri à Bordeaux-Bègles. Le dispositif sera déployé sur les vingt-sept postes de tri que compte la cabine de tri. La mise en service du centre de tri est prévue pour fin 2021.

Augmentation du nombre d'indicateurs mesurés Une troisième perspective consiste à améliorer le dispositif de mesure des indicateurs. Ainsi, il est envisagé d'effectuer la mesure de nouveaux indicateurs en utilisant toutes les articulations de notre modèle d'estimation de posture, permettant par exemple de mesurer l'orientation du visage du valoriste et donc sa capacité d'anticipation de la matière à trier, ou bien de mesurer l'effort de préhension que nécessite son activité de tri.

Généralisation du dispositif À l'heure actuelle, nous considérons que chaque dispositif installé sur un poste de tri nécessite un entraînement séparé et indépendant des autres dispositifs. Cependant, avec l'acquisition de données issues d'un grand nombre de postes, nous pourrons dans un deuxième temps chercher à mettre en place un entraînement permettant d'utiliser le même modèle sur tous les postes, et ne nécessitant donc plus d'entraînement au moment du déploiement, mais une maintenance à partir des données acquises au fil du temps.

2.2 Perspectives de recherche

Les travaux menés pendant cette thèse ouvrent naturellement de nombreuses perspectives de recherche. On distingue trois grandes perspectives de recherche : l'extension de nos travaux à de nouveaux types de données, tels que les nuages de points, l'utilisation de l'information temporelle pour le traitement de séquences d'images de profondeur et l'extension à d'autres modèles d'apprentissage profond.

Les images de profondeur vues comme nuages de points Les images de profondeur, par leur nature, peuvent facilement être transformées en nuage de points, lorsqu'on connaît les paramètres du capteur qui a permis leur acquisition. Ceci permettrait d'utiliser une réelle information 3D pour la segmentation de l'opérateur dans l'image et pour l'estimation de sa posture. Les méthodes d'apprentissage proposées pour les images de profondeur devront alors être étendues aux nuages de points. Nous avons déjà réalisé quelques expériences encourageantes avec le modèle PointNet [Qi et al., 2017b].

Utilisation de l'information temporelle Dans nos travaux, chaque image de profondeur est traitée de manière indépendante. Dans la réalité, nous acquérons des séquences vidéo. Ainsi, il existe une information temporelle qui n'est pas encore utilisée. Étendre nos modèles pour effectuer la segmentation de séquences d'images ou l'estimation de posture 3D sur des séquences peut permettre d'augmenter la robustesse des algorithmes développés.

Utilisation de données annotées Les tests en conditions réelles, dont la réalisation est prévue dans les prochains mois, peuvent être l'occasion d'acquérir un petit ensemble de données annotées. Ces données pourraient alors être intégrées à un entraînement de nos modèles pour améliorer les performances de notre dispositif.

Traduction d'image à image comme augmentation de données Le processus de traduction d'image à image nous permet d'altérer nos données et d'en acquérir de nouvelles. Il peut donc être vu comme un outil d'augmentation de données. Les données nouvellement construites peuvent alors être associées aux données d'origine pour entraîner un

modèle qui serait alors plus robuste. Ce processus d'augmentation de données, basé sur des données existantes mais non supervisé, pourrait de manière générale être utilisé pou n'importe quel problème comme un moyen de régulariser un peu plus des modèles très profonds. De même, plusieurs modèles de traduction d'image à image permettraient de générer plus de données, qui pourraient être utilisées ensemble.

Annexe

Calcul des angulations et des indicateurs

Dans cette annexe, nous expliquons le procédé de calcul des angulations et des indicateurs ergonomiques, c'est-à-dire le troisième module du pipeline, illustré en Figure 1. Les articulations utilisées pour calculer ces indicateurs sont données en Figure 2.

L'angle, exprimé en radians, entre deux vecteurs $\overrightarrow{u}$ et $\overrightarrow{v}$ peut être obtenu par la formule suivante :

$$(\widehat{\overrightarrow{u}, \overrightarrow{v}}) = \arctan\left(\overrightarrow{\overrightarrow{u} \times \overrightarrow{v}}, \overrightarrow{u} \cdot \overrightarrow{v}\right) \tag{11}$$

1 Zones des angulations

Le calcul d'une angulation est réalisé à l'aide de la position des différentes articulations concernées par cette angulation. Si nécessaire, et en fonction de l'angulation à calculer, ces articulations sont projetées sur un plan. Le calcul de chaque angulation est détaillé ici.

Flexion et extension du cou Pour calculer cette angulation, on utilise les articulations de la tête, du cou et du haut /de la colonne vertébrale. L'angle obtenu permet de mesurer la flexion ou l'extension du cou. Une illustration est proposée en Figure 3, à gauche.

Rotation du cou Pour calculer cette angulation, on se sert des articulations de la tête, du nez et des deux épaules. L'angle entre le vecteur de la tête au nez et le vecteur formé par les articulations des épaules permet d'obtenir la rotation du cou. Une illustration est proposée en Figure 3, à droite.

Flexion vers l'avant du buste Cette angulation est calculée à l'aide des articulations du bas et du haut de la colonne vertébrale. On construit un plan dont le vecteur normal est le vecteur entre les deux épaules. En mesurant l'angle formé par le vecteur ayant pour extrémités la projection des deux articulations et un vecteur unitaire orienté vers le haut sur ce plan, on obtient la flexion vers l'avant du buste. Une illustration est proposée en Figure 4, à gauche.

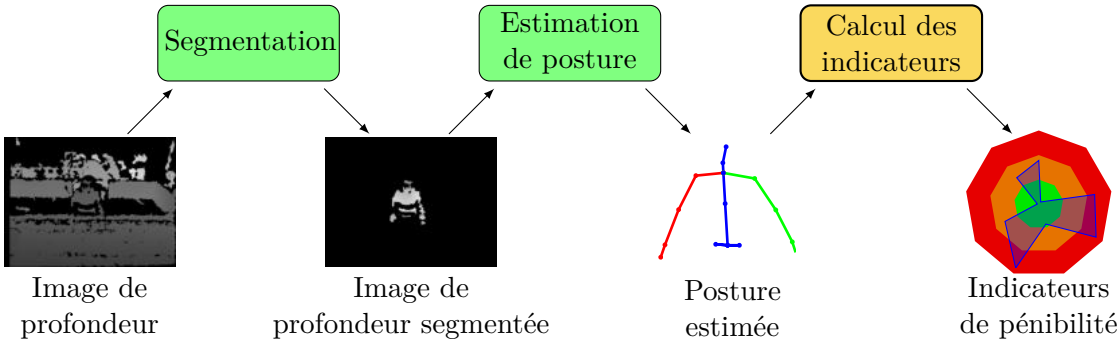


FIGURE 1 – Notre pipeline pour le calcul des indicateurs de pénibilité à partir des images de profondeur. Dans ce chapitre, nous considérons les deux premiers modules terminés (en vert), et nous nous concentrons sur le troisième module (en jaune), qui nous permettra de calculer les indicateurs à partir de l'estimation de la posture 3D de l'opérateur humain.

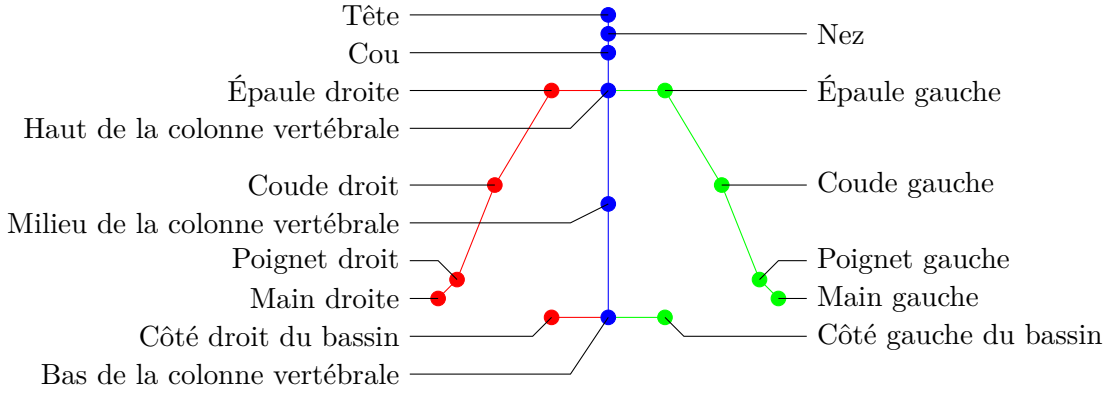


FIGURE 2 – Les seize articulations utilisées pour calculer les angulations et les indicateurs ergonomiques.

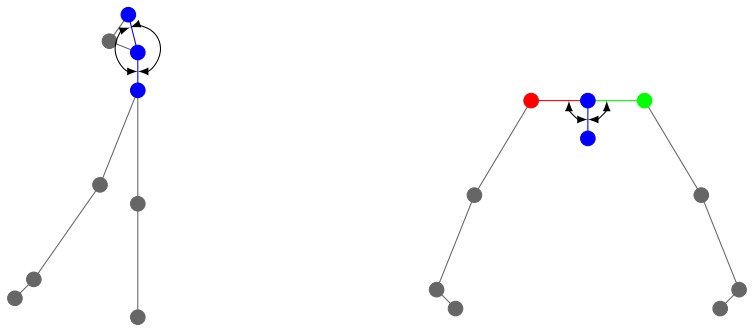
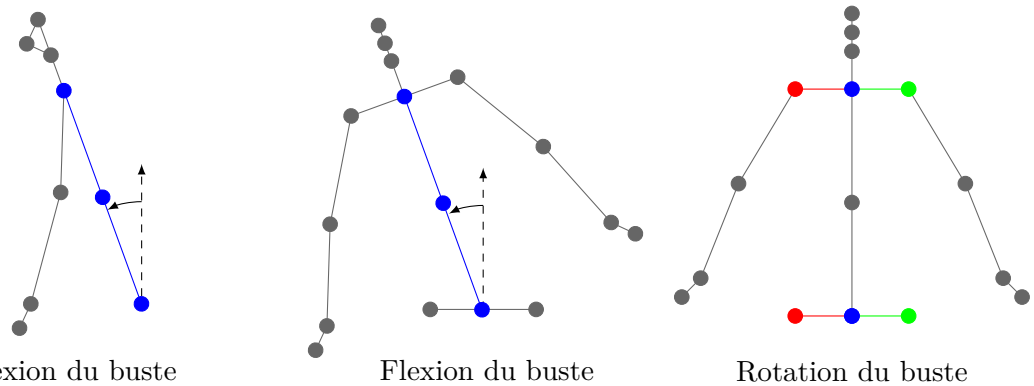


FIGURE 3 – Articulations utilisées pour le calcul de la flexion et de l'extension du cou (à gauche) et de sa rotation (à droite).



Flexion du buste vers l'avant (vue de côté) | Flexion du buste sur le côté (vue de face) | Rotation du buste

FIGURE 4 – Articulations utilisées pour le calcul de la flexion vers l'avant (à gauche) et sur le côté (au milieu) du buste, et de sa rotation (à droite).

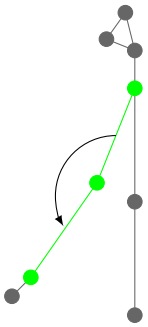
Flexion sur le côté du buste Cette angulation est calculée à l'aide des articulations du bas et du haut de la colonne vertébrale. On construit le plan orthogonal au plan précédent ayant le vecteur unitaire orienté vers le haut comme vecteur directeur. En mesurant l'angle formé par le vecteur ayant pour extrémités la projection des deux articulations sur ce plan et le vecteur unitaire orienté vers le haut, on obtient la flexion sur le côté du buste. Une illustration est proposée en Figure 4, au milieu.

Rotation du buste Pour calculer cette angulation, on se sert des articulations des épaules et du bassin. En mesurant l'angle formé par le vecteur ayant pour extrémités les articulations des épaules et le vecteur ayant pour extrémités les articulations du bassin, on obtient la rotation du buste. Une illustration est proposée en Figure 4, à droite.

Flexion du coude Cette angulation est obtenue en calculant l'angle formé par les articulations de l'épaule, du coude et du poignet du bras concerné. Une illustration est proposée en Figure 5.

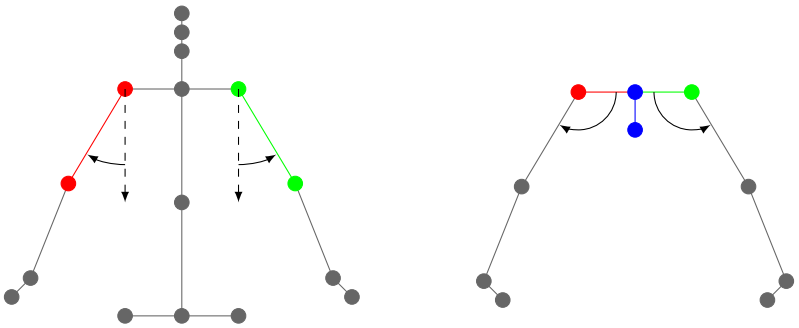
Élévation de l'épaule Cette angulation est obtenue en calculant l'angle formé par le vecteur ayant pour extrémités l'épaule et le coude du bras concerné et le vecteur unitaire vertical orienté vers le bas. Une illustration est proposée en Figure 6, à gauche.

Rotation de l'épaule Cette angulation est calculée à l'aide de l'articulation du coude du bras concerné et des articulations des deux épaules. On construit un plan ayant pour vecteur normal le vecteur formé par les articulations du haut et du bas de la colonne vertébrale, puis on projette les trois articulations citées précédemment sur ce plan. L'angle



Flexion du coude
(vue de côté)

FIGURE 5 – Articulations utilisées pour le calcul de la flexion du coude.



Élévation des épaules Rotation des épaules

FIGURE 6 – Articulations utilisées pour le calcul de l'élévation (à gauche) et de la rotation (à droite) des épaules.

formé par la projection des ces trois articulations donne l'angle de rotation de l'épaule du bras concerné. Une illustration est proposée en Figure 6, à droite.

Une fois la valeur d'angulation obtenue, la zone d'appartenance de l'angulation est ensuite directement évaluée en fonction de la valeur de l'angulation et des seuils de séparation des zones prédéfinies (voir Tableau 4).

2 Zones d'évolution des mains

La zone d'évolution d'une main est obtenue d'une manière similaire : tout d'abord, la distance par rapport au tronc de l'opérateur est calculée, et peut déjà permettre une classification : hors zone si elle est supérieure à 0,8 fois la taille du bras (distance cumulée épaule-coude-poignet-main) ou zone rouge, si elle est supérieure à 0,7 fois la taille du bras. Si la distance entre le tronc de l'opérateur et la main est inférieure à 0,7 fois la taille du bras, on calcule des droites séparatrices des différentes zones ergonomiques sur un plan

Zone verte : risque réduit Zone jaune : risque accru Zone rouge : risque critique

Articulation	Angulation	Zones
Cou	Extension	Inférieure à 20° Comprise entre 20° et 40° Supérieure à 40°
	Flexion	Inférieure à 20° Comprise entre 20° et 40° Supérieure à 40°
	Rotation	Inférieure à 10° Comprise entre 10° et 55° Supérieure à 55°
Buste/Dos	Flexion vers l'avant	Inférieure à 20° Comprise entre 20° et 45° Supérieure à 45°
	Flexion sur le côté	Aucune flexion Inférieure à 10° Supérieure à 10°
	Rotation	Aucune rotation Inférieure à 40° Supérieure à 40°
Coudes	Flexion	Nulle (0°) Comprise entre 0° et 75° Comprise entre 75° et 140° Supérieure à 140°
Épaules	Élévation	Inférieure à 90° Comprise entre 90°et 120° Supérieure à 120°
	Rotation	Inférieure à 80° Comprise entre 80° et 90° Supérieure à 90°

TABLE 4 – Les zones de risque pour les angulations de chaque articulation, définies d'après l'étude ergonomique. Les zones sont données par la couleur des angulations.

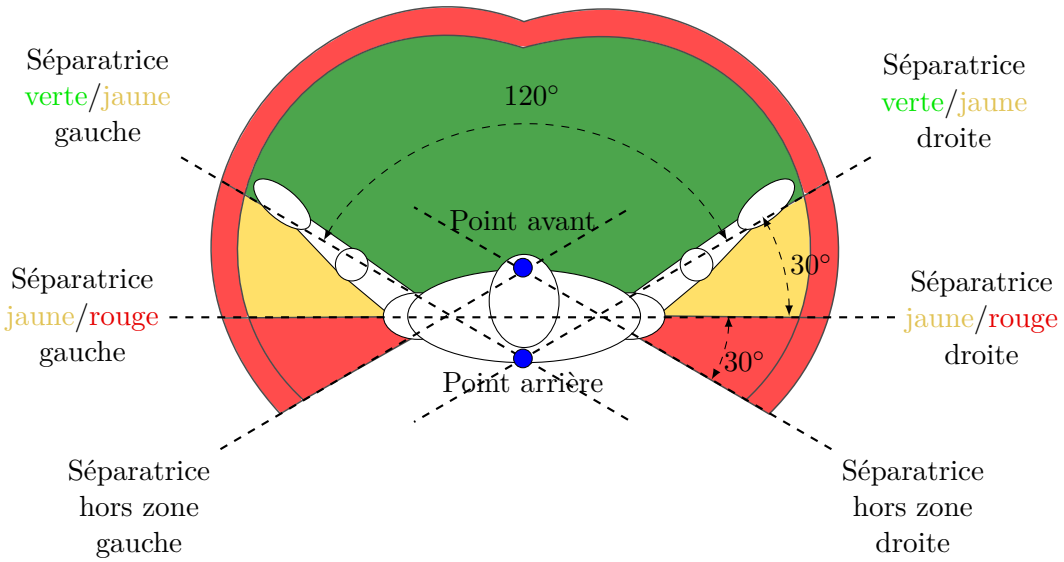


FIGURE 7 – Les différentes zones autour d'un valoriste, telles que définies par la norme *NF X35—702* : zone verte (risque réduit), zone jaune (risque accru), zone rouge (risque critique), hors zone (risque délétère). On a ajouté les outils géométriques utilisés pour calculer ces zones d'évolution, lorsqu'on connaît la position des mains.

horizontal passant par l'articulation du côté du bassin de la main dont on souhaite calculer la zone d'évolution. La droite de séparation entre la zone verte et jaune est calculée à l'aide d'un point, situé à l'arrière de l'opérateur et équidistant des deux articulations des côtés du bassin, formant un triangle isocèle dont les angles à la base sont de 30° : l'angle au niveau du point est de 120°, comme voulu par la norme (voir Section 1.3.2). La droite de séparation entre la zone jaune et la zone rouge est la droite du plan horizontal passant par les deux articulations des côtés du bassin, et l'angle des zones jaunes mesure 30°. La droite de séparation de la zone rouge et d'une position hors-zone est calculée de manière analogue, à la droite de séparation de la zone verte et de la zone jaune : à l'aide d'un point, situé cette fois à l'avant de l'opérateur et équidistant des deux articulations des côtés du bassin, formant un triangle isocèle dont les angles à la base sont de 30°. En fonction de la position de la main par rapport à ces droites, on détermine alors sa zone d'évolution. Les différents points et droites utilisés pour calculer les zones d'évolution de la main sont donnés en Figure 7.

Le calcul des indicateurs finaux, qui reste à implanter, sera effectué en agrégeant les différentes zones calculées sur une séquence d'une durée déterminée.

Bibliographie

[Adams and Bischof, 1994] Adams, R. and Bischof, L. (1994). Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6) :641–647.

[AFNOR, 2008] AFNOR (2008). Sécurité des machines - performance physique humaine - partie 4 : évaluation des postures et mouvements lors du travail en relation avec les machines. Association française de normalisation.

[AFNOR, 2015] AFNOR (2015). Sécurité des machines — principes ergonomiques pour la conception des cabines de tri manuel des déchets recyclables secs ménagers et assimilés issus des collectes sélectives. Association française de normalisation.

[Agarwal and Triggs, 2004a] Agarwal, A. and Triggs, B. (2004a). 3d human pose from silhouettes by relevance vector regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 882–888. IEEE.

[Agarwal and Triggs, 2004b] Agarwal, A. and Triggs, B. (2004b). Learning to track 3d human motion from silhouettes. In *Proceedings of the 21st International Conference on Machine Learning*, page 2.

[Agarwal and Triggs, 2005] Agarwal, A. and Triggs, B. (2005). Recovering 3d human pose from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1) :44–58.

[Agarwal and Triggs, 2006] Agarwal, A. and Triggs, B. (2006). A local basis representation for estimating human pose from cluttered images. In *Asian Conference on Computer Vision*, pages 50–59. Springer.

[Aggarwal and Xia, 2014] Aggarwal, J. K. and Xia, L. (2014). Human activity recognition from 3d data : A review. *Pattern Recognition Letters*, 48 :70–80.

[Akhter and Black, 2015] Akhter, I. and Black, M. J. (2015). Pose-conditioned joint angle limits for 3d human pose reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1446–1455. IEEE.

[AlBahar and Huang, 2019] AlBahar, B. and Huang, J.-B. (2019). Guided image-to-image translation with bi-directional feature transformation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9016–9025. IEEE.

[Albiol et al., 2001] Albiol, A., Torres, L., and Delp, E. J. (2001). An unsupervised color

image segmentation algorithm for face detection applications. In *Proceedings of the IEEE International Conference on Image Processing*, volume 2, pages 681–684. IEEE.

[Almahairi et al., 2018] Almahairi, A., Rajeshwar, S., Sordoni, A., Bachman, P., and Courville, A. (2018). Augmented cyclegan : Learning many-to-many mappings from unpaired data. In *Proceedings of the 35th International Conference on Machine Learning*, pages 195–204.

[Alsmirat et al., 2017] Alsmirat, M. A., Jararweh, Y., Al-Ayyoub, M., Shehab, M. A., and Gupta, B. B. (2017). Accelerating compute intensive medical imaging segmentation algorithms using hybrid cpu-gpu implementations. *Multimedia Tools and Applications*, 76(3) :3537–3555.

[Ambrosio and Tortorelli, 1990] Ambrosio, L. and Tortorelli, V. M. (1990). Approximation of functional depending on jumps by elliptic functional via t-convergence. *Communications on Pure and Applied Mathematics*, 43(8) :999–1036.

[Amit and Geman, 1997] Amit, Y. and Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural computation*, 9(7) :1545–1588.

[Andriluka et al., 2009] Andriluka, M., Roth, S., and Schiele, B. (2009). Pictorial structures revisited : People detection and articulated pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1014–1021. IEEE.

[Andriluka et al., 2010] Andriluka, M., Roth, S., and Schiele, B. (2010). Monocular 3d pose estimation and tracking by detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 623–630. IEEE.

[Andriluka et al., 2014] Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2d human pose estimation : New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3686–3693.

[Anguelov et al., 2005] Anguelov, D., Srinivasan, P., Koller, D., Thrun, S., Rodgers, J., and Davis, J. (2005). Scape : Shape completion and animation of people. *ACM Transactions on Graphics*, pages 408–416.

[Aptel and Cnockaert, 2002] Aptel, M. and Cnockaert, J. C. (2002). Liens entre les troubles musculo-squelettiques du membre supérieur et le stress. *BTS, le stress au travail*, 19-20 :57–63.

[Arnold et al., 2007] Arnold, A., Nallapati, R., and Cohen, W. (2007). A comparative study of methods for transductive transfer learning. In *Proceedings of the 7th International Conference on Data Mining Workshops*, pages 77–82. IEEE.

[Aydin et al., 2017] Aydin, I., Karakose, M., Hamsin, G. G., Sarimaden, A., and Akin, E. (2017). A new object detection and classification method for quality control based on segmentation and geometric features. In *International Artificial Intelligence and Data Processing Symposium*, pages 1–6. IEEE.

[Baak et al., 2013] Baak, A., Müller, M., Bharaj, G., Seidel, H.-P., and Theobalt, C. (2013). A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Consumer Depth Cameras for Computer Vision*, Advances in Computer Vision and Pattern Recognition, pages 71–98. Springer.

[Babenko, 2008] Babenko, B. (2008). Multiple instance learning : Algorithms and applications. *University of California San Diego Computer Science and Engineering Department*.

[Badrinarayanan et al., 2017] Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). Segnet : A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39 :2481–2495.

[Balan et al., 2007] Balan, A. O., Sigal, L., Black, M. J., Davis, J. E., and Haussecker, H. W. (2007). Detailed human shape and pose from images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Baldi, 2012] Baldi, P. (2012). Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of 28th International Conference on Machine Learning Workshop on Unsupervised and Transfer Learning*, pages 37–49.

[Banerjee et al., 2012] Banerjee, T., Rantz, M., Li, M., Popescu, M., Stone, E., Skubic, M., and Scott, S. (2012). Monitoring hospital rooms for safety using depth images. In *Proceedings of the AAAI Fall Symposium Series AI for Gerontechnology*. Citeseer.

[Barrón and Kakadiaris, 2001] Barrón, C. and Kakadiaris, I. A. (2001). Estimating anthropometry and pose from a single uncalibrated image. *Computer Vision and Image Understanding*, 81(3) :269–284.

[Belagiannis and Zisserman, 2017] Belagiannis, V. and Zisserman, A. (2017). Recurrent human pose estimation. In *Proceeding of the 12th IEEE International Conference on Automatic Face & Gesture Recognition*, pages 468–475. IEEE.

[Belongie et al., 2001] Belongie, S., Malik, J., and Puzicha, J. (2001). Shape context : A new descriptor for shape matching and object recognition. In *Advances in Neural Information Processing Systems*, pages 831–837.

[Belongie et al., 2002] Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4) :509–522.

[Bengio, 2012] Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. In *Proceedings of the 29th International Conference on Machine Learning Workshop on Unsupervised and Transfer Learning Workshop*, volume 27, pages 17–37.

[Besl and McKay, 1992] Besl, P. J. and McKay, N. D. (1992). A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2) :239–256.

[Beucher and Lantuéjoul, 1979] Beucher, S. and Lantuéjoul, C. (1979). Use of watersheds in contour detection. In *Proceedings of the International Workshop on Image Processing*.

[Beucher, 1992] Beucher, S. (1992). The watershed transformation applied to image segmentation. *Scanning Microscopy - Supplement*, pages 299–299.

[Bishop et al., 1995] Bishop, C. M. et al. (1995). *Neural networks for pattern recognition*. Oxford University Press.

[Blanc-Beyne et al., 2019] Blanc-Beyne, T., Carlier, A., and Charvillat, V. (2019). Iterative dataset filtering for weakly supervised segmentation of depth images. In *Proceedings of the IEEE International Conference on Image Processing*, pages 1515–1519. IEEE.

[Blender Online Community, 2020] Blender Online Community (2020). *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam.

[Blum and Mitchell, 1998] Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the Annual ACM Conference on Computational Learning Theory*, pages 92–100. Association for Computing Machinery.

[Bogo et al., 2016] Bogo, F., Kanazawa, A., Lassner, C., Gehler, P., Romero, J., and Black, M. J. (2016). Keep it smpl : Automatic estimation of 3d human pose and shape from a single image. In *Proceedings of the European Conference on Computer Vision*, pages 561–578. Springer.

[Bourdev and Malik, 2009] Bourdev, L. and Malik, J. (2009). Poselets : Body part detectors trained using 3d human pose annotations. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1365–1372. IEEE.

[Boykov and Funka-Lea, 2006] Boykov, Y. and Funka-Lea, G. (2006). Graph cuts and efficient nd image segmentation. *International Journal of Computer Vision*, 70(2) :109–131.

[Boykov and Jolly, 2001] Boykov, Y. Y. and Jolly, M.-P. (2001). Interactive graph cuts for optimal boundary & region segmentation of objects in nd images. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1, pages 105–112. IEEE.

[Brand, 1999] Brand, M. (1999). Shadow puppetry. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, pages 1237–1244. IEEE.

[Brazil et al., 2017] Brazil, G., Yin, X., and Liu, X. (2017). Illuminating pedestrians via simultaneous detection & segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4950–4959. IEEE.

[Bregler and Malik, 1998] Bregler, C. and Malik, J. (1998). Tracking people with twists and exponential maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8–15. IEEE.

[Breiman, 1996] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2) :123–140.

[Brière et al., 2015] Brière, J., Fouquet, N., Ha, C., Imbernon, E., Plaine, J., Rivière, S., Roquelaure, Y., and Valenty, M. (2015). Des indicateurs en santé travail. *Image*, 4 :48.

[Brodley and Friedl, 1999] Brodley, C. E. and Friedl, M. A. (1999). Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11 :131–167.

[Bryant and Pettigrew, 2015] Bryant, A. E. and Pettigrew, D. (2015). Method and ui for z depth image segmentation. US Patent App. 14/053,581.

[Bulat and Tzimiropoulos, 2016] Bulat, A. and Tzimiropoulos, G. (2016). Human pose estimation via convolutional part heatmap regression. In *Proceedings of the European Conference on Computer Vision*, pages 717–732. Springer.

[Burnham and Anderson, 2002] Burnham, K. P. and Anderson, D. R. (2002). *Model selection and multimodel inference*. Springer.

[Buscemi et al., 2019] Buscemi, V., Chang, W.-J., Liston, M. B., McAuley, J. H., and Schabrun, S. M. (2019). The role of perceived stress and life stressors in the development of chronic musculoskeletal pain disorders : A systematic review. *The Journal of Pain*.

[Butler et al., 2012] Butler, D. A., Izadi, S., Hilliges, O., Molyneaux, D., Hodges, S., and Kim, D. (2012). Shake'n'sense : reducing interference for overlapping structured light depth cameras. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1933–1936.

[Byeon et al., 2015] Byeon, W., Breuel, T. M., Raue, F., and Liwicki, M. (2015). Scene labeling with lstm recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3547–3555.

[Canny, 1986] Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8 :679–698.

[Cao et al., 2017] Cao, Z., Simon, T., Wei, S.-E., and Sheikh, Y. (2017). Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7291–7299. IEEE.

[Cao et al., 2018] Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E., and Sheikh, Y. (2018). Openpose : realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv :1812.08008*.

[Carreira et al., 2016] Carreira, J., Agrawal, P., Fragkiadaki, K., and Malik, J. (2016). Human pose estimation with iterative error feedback. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4733–4742. IEEE.

[Carsat HF, 2020] Carsat HF (2020). Connaître les tms et les enjeux de la prévention. Caisse d'assurance retraite et de la santé au travail - Hauts-de-France.

[Carsat PL, 2020] Carsat PL (2020). Troubles musculo-squelettiques (tms). Caisse d'assurance retraite et de la santé au travail - Pays de la Loire.

[Carson et al., 2002] Carson, C., Belongie, S., Greenspan, H., and Malik, J. (2002). Blobworld : Image segmentation using expectation-maximization and its application to image querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8) :1026–1038.

[Caruana, 1997] Caruana, R. (1997). Multitask learning. *Machine Learning*, 28 :41–75.

[Chen et al., 2011] Chen, C., Yang, Y., Nie, F., and Odobez, J.-M. (2011). 3d human pose recovery from image by efficient visual feature selection. *Computer Vision and Image Understanding*, 115(3) :290–299.

[Chen et al., 2016a] Chen, C., Liu, K., and Kehtarnavaz, N. (2016a). Real-time human action recognition based on depth motion maps. *Journal of Real-Time Image Processing*, 12 :155–163.

[Chen and Ramanan, 2017] Chen, C.-H. and Ramanan, D. (2017). 3d human pose estimation= 2d pose estimation+ matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7035–7043. IEEE.

[Chen et al., 2013] Chen, L., Wei, H., and Ferryman, J. (2013). A survey of human motion analysis using depth imagery. *Pattern Recognition Letters*, 34(15) :1995–2006.

[Chen et al., 2015] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2015). Semantic image segmentation with deep convolutional nets and fully connected crfs. In *International Conference on Learning Representations*.

[Chen et al., 2017a] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2017a). Deeplab : Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4) :834–848.

[Chen et al., 2017b] Chen, L.-C., Papandreou, G., Schroff, F., and Adam, H. (2017b). Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv :1706.05587*.

[Chen et al., 2016b] Chen, W., Wang, H., Li, Y., Su, H., Wang, Z., Tu, C., Lischinski, D., Cohen-Or, D., and Chen, B. (2016b). Synthesizing training images for boosting human 3d pose estimation. In *4th International Conference on 3D Vision*, pages 479–488. IEEE.

[Chen and Yuille, 2014] Chen, X. and Yuille, A. L. (2014). Articulated pose estimation by a graphical model with image dependent pairwise relations. In *Advances in Neural Information Processing Systems*, pages 1736–1744.

[Chen and Yuille, 2015] Chen, X. and Yuille, A. L. (2015). Parsing occluded people by flexible compositions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3945–3954.

[Chen and Medioni, 1992] Chen, Y. and Medioni, G. G. (1992). Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3) :145–155.

[Chen et al., 2017c] Chen, Y., Shen, C., Wei, X.-S., Liu, L., and Yang, J. (2017c). Adversarial posenet : A structure-aware convolutional network for human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1212–1221.

[Chen et al., 2018a] Chen, Y., Lai, Y.-K., and Liu, Y.-J. (2018a). Cartoongan : Generative adversarial networks for photo cartoonization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9465–9474. IEEE.

[Chen et al., 2018b] Chen, Y., Wang, Z., Peng, Y., Zhang, Z., Yu, G., and Sun, J. (2018b). Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7103–7112.

[Cheng and Butler, 2005] Cheng, H. and Butler, D. (2005). Segmentation of aerial surveillance video using a mixture of experts. In *Proceedings of the International Conference on Digital Image Computing : Techniques and Applications*, pages 66–66. IEEE.

[Cheng, 1995] Cheng, Y. (1995). Mean shift, mode seeking, and clustering. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, volume 17, pages 790–799. IEEE.

[Chéron et al., 2015] Chéron, G., Laptev, I., and Schmid, C. (2015). P-cnn : Pose-based cnn features for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3218–3226. IEEE.

[Choi et al., 2018] Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., and Choo, J. (2018). Stargan : Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE.

[Chou et al., 2018] Chou, E., Tan, M., Zou, C., Guo, M., Haque, A., Milstein, A., and Fei-Fei, L. (2018). Privacy-preserving action recognition for smart hospitals using low-resolution depth images. *arXiv preprint arXiv :1811.09950*.

[Chu et al., 2016] Chu, X., Ouyang, W., Li, H., and Wang, X. (2016). Structured feature learning for pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4715–4723.

[Chu et al., 2017] Chu, X., Yang, W., Ouyang, W., Ma, C., Yuille, A. L., and Wang, X. (2017). Multi-context attention for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1831–1840. IEEE.

[Cireşan et al., 2010] Cireşan, D. C., Meier, U., Gambardella, L. M., and Schmidhuber, J. (2010). Deep, big, simple neural nets for handwritten digit recognition. *Neural Computation*, 22(12) :3207–3220.

[Cireşan et al., 2012] Cireşan, D. C., Giusti, A., Gambardella, L. M., and Schmidhuber, J. (2012). Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in Neural Information Processing Systems*, pages 2843–2851.

[Citeo et Adelphe, 2018] Citeo et Adelphe (2018). Rapport d'activité Citeo et Adelphe. Citeo et Adelphe.

[CNAM, 2009] CNAM (2009). Prévention des tms : exemples de réalisations. Caisse Nationale de l'Assurance Maladie.

[CNAM, 2018] CNAM (2018). L'essentiel 2018 : Santé et sécurité au travail. Caisse Nationale de l'Assurance Maladie.

[CNIL, 2019] CNIL (2019). Loi "informatique et libertés". Commission Nationale de l'Informatique et des Libertés.

[Comaniciu and Meer, 2002] Comaniciu, D. and Meer, P. (2002). Mean shift : A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5) :603–619.

[Corazza et al., 2006] Corazza, S., Muendermann, L., Chaudhari, A., Demattio, T., Cobelli, C., and Andriacchi, T. P. (2006). A markerless motion capture system to study musculoskeletal biomechanics : visual hull and simulated annealing approach. *Annals of Biomedical Engineering*, 34(6) :1019–1029.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3) :273–297.

[Covell et al., 2006] Covell, M. M., Lin, M. H., Rahimi, A., Harville, M., Darrell, T. J., Woodfill, J. I., Baker, H., and Gordon, G. G. (2006). Three dimensional object pose estimation which employs dense depth information. US Patent 7,003,134.

[Csurka et al., 2004] Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C. (2004). Visual categorization with bags of keypoints. In *Proceedings of the European Conference on Computer Vision Workshop on Statistical Learning in Computer Vision*, volume 1, pages 1–2. Springer.

[Cybenko, 1989] Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2 :303–314.

[Dahl et al., 2011] Dahl, G. E., Yu, D., Deng, L., and Acero, A. (2011). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *Transactions on Audio, Speech, and Language Processing*, 20(1) :30–42.

[Dai et al., 2008] Dai, W., Yang, Q., Xue, G.-R., and Yu, Y. (2008). Self-taught clustering. In *Proceedings of the 25th International Conference on Machine Learning*, pages 200–207. Association for Computing Machinery.

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 886–893. IEEE.

[Dantone et al., 2013] Dantone, M., Gall, J., Leistner, C., and Van Gool, L. (2013). Human pose estimation using body parts dependent joint regressors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3041–3048.

[Dasgupta and Hsu, 2008] Dasgupta, S. and Hsu, D. (2008). Hierarchical sampling for active learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 208–215. Association for Computing Machinery.

[De Aguiar et al., 2007] De Aguiar, E., Theobalt, C., Stoll, C., and Seidel, H.-P. (2007). Marker-less deformable mesh tracking for human shape and motion capture. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Delalande-Danet et al., 2015] Delalande-Danet, V., Desarmenien, A., Incorvaïa, A.-M., Letheux, C., Leviel, C., and Viossat, M. (2015). Guide des troubles musculosquelettiques. Cisme - Présanse.

[Delamarre and Faugeras, 2001] Delamarre, Q. and Faugeras, O. (2001). 3d articulated models and multiview tracking with physical forces. *Computer Vision and Image Understanding*, 81(3) :328–357.

[Delingette and Montagnat, 2001] Delingette, H. and Montagnat, J. (2001). Shape and topology constraints on parametric active contours. *Computer Vision and Image Understanding*, 83(2) :140–171.

[Demirdjian et al., 2003] Demirdjian, D., Ko, T., and Darrell, T. (2003). Constraining human body tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, page 1071. IEEE.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society : Series B (Methodological)*, 39(1) :1–22.

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet : A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE.

[Denton et al., 2015] Denton, E. L., Chintala, S., Fergus, R., et al. (2015). Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in Neural Information Processing Systems*, pages 1486–1494.

[Deutscher et al., 2000] Deutscher, J., Blake, A., and Reid, I. (2000). Articulated body motion capture by annealed particle filtering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 126–133. IEEE.

[Dhankhar and Sahu, 2013] Dhankhar, P. and Sahu, N. (2013). A review and research of edge detection techniques for image segmentation. *International Journal of Computer Science and Mobile Computing*, 2(7) :86–92.

[Dietterich et al., 1997] Dietterich, T. G., Lathrop, R. H., and Lozano-Pérez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89 :31–71.

[Dimitrijevic et al., 2006] Dimitrijevic, M., Lepetit, V., and Fua, P. (2006). Human body pose detection using bayesian spatio-temporal templates. *Computer Vision and Image Understanding*, 104 :127–139.

[Droeschel and Behnke, 2011] Droeschel, D. and Behnke, S. (2011). 3d body pose estimation using an adaptive person model for articulated icp. In *International Conference on Intelligent Robotics and Applications*, pages 157–167. Springer.

[Dutrieux et al., 2018] Dutrieux, A., Gaillard, I., Mollo, V., Blanc-Beyne, T., Carlier, A., and Charvillat, V. (2018). De l'actimétrie à l'activité, quels sont les apports de l'ergonome à la conception d'un outil numérique de mesure sur le travail ? In *L'ergonomie à quelles échelles ?*, pages 322–327. Société d'ergonomie de langue française.

[Eigen and Fergus, 2015] Eigen, D. and Fergus, R. (2015). Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2650–2658.

[Elgammal and Lee, 2004] Elgammal, A. and Lee, C.-S. (2004). Inferring 3d body pose from silhouettes using activity manifold learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 681–688. IEEE.

[Erhan et al., 2009] Erhan, D., Manzagol, P.-A., Bengio, Y., Bengio, S., and Vincent, P. (2009). The difficulty of training deep architectures and the effect of unsupervised pretraining. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics*, pages 153–160.

[Espindola et al., 2006] Espindola, G., Câmara, G., Reis, I., Bins, L., and Monteiro, A. (2006). Parameter selection for region-growing image segmentation algorithms using spatial autocorrelation. *International Journal of Remote Sensing*, 27(14) :3035–3040.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, volume 96, pages 226–231. AAAI Press.

[Everitt and Skrondal, 2006] Everitt, B. and Skrondal, A. (2006). *The Cambridge dictionary of statistics*. Cambridge University Press.

[Fan et al., 2015] Fan, X., Zheng, K., Lin, Y., and Wang, S. (2015). Combining local appearance and holistic view : Dual-source deep neural networks for human pose es-

timation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1347–1355.

[Fang et al., 2017] Fang, H.-S., Xie, S., Tai, Y.-W., and Lu, C. (2017). Rmpe : Regional multi-person pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2334–2343.

[Farabet et al., 2012] Farabet, C., Couprie, C., Najman, L., and LeCun, Y. (2012). Scene parsing with multiscale feature learning, purity trees, and optimal covers. In *Proceedings of the 29th International Conference on Machine Learning*, pages 1857–1864.

[Farabet et al., 2013] Farabet, C., Couprie, C., Najman, L., and LeCun, Y. (2013). Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8) :1915–1929.

[Felzenszwalb and Huttenlocher, 2005] Felzenszwalb, P. F. and Huttenlocher, D. P. (2005). Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1) :55–79.

[Felzenszwalb et al., 2009] Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2009). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9) :1627–1645.

[Fenster and Chiu, 2006] Fenster, A. and Chiu, B. (2006). Evaluation of segmentation algorithms for medical imaging. In *Proceedings of the IEEE International Conference on Engineering in Medicine and Biology Society*, pages 7186–7189. IEEE.

[Ferrari et al., 2008] Ferrari, V., Marin-Jimenez, M., and Zisserman, A. (2008). Progressive search space reduction for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Fischler and Elschlager, 1973] Fischler, M. A. and Elschlager, R. A. (1973). The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 100(1) :67–92.

[Freifeld et al., 2010] Freifeld, O., Weiss, A., Zuffi, S., and Black, M. J. (2010). Contour people : A parameterized model of 2d articulated human shape. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 639–646. IEEE.

[Freund and Schapire, 1995] Freund, Y. and Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37. Springer.

[Gall et al., 2010] Gall, J., Yao, A., and Van Gool, L. (2010). 2d action recognition serves 3d human pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 425–438. Springer.

[Ganapathi et al., 2010] Ganapathi, V., Plagemann, C., Koller, D., and Thrun, S. (2010). Real time motion capture using a single time-of-flight camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 755–762. IEEE.

[Ganapathi et al., 2012] Ganapathi, V., Plagemann, C., Koller, D., and Thrun, S. (2012). Real-time human pose tracking from range data. In *Proceedings of the European conference on computer vision*, pages 738–751. Springer.

[Ganin et al., 2016] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17 :2096–2030.

[Gatys et al., 2016] Gatys, L. A., Ecker, A. S., and Bethge, M. (2016). Image style transfer using convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423. IEEE.

[Gatys et al., 2017] Gatys, L. A., Ecker, A. S., Bethge, M., Hertzmann, A., and Shechtman, E. (2017). Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3985–3993. IEEE.

[Gavrila and Davis, 1996] Gavrila, D. M. and Davis, L. S. (1996). 3-d model-based tracking of humans in action : a multi-view approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 73–80. IEEE.

[Girshick et al., 2011a] Girshick, R., Shotton, J., Kohli, P., Criminisi, A., and Fitzgibbon, A. (2011a). Efficient regression of general-activity human poses from depth images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 415–422. IEEE.

[Girshick et al., 2015] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2015). Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1) :142–158.

[Girshick et al., 2011b] Girshick, R. B., Felzenszwalb, P. F., and Mcallester, D. A. (2011b). Object detection with grammar models. In *Advances in Neural Information Processing Systems*, pages 442–450.

[Glorot et al., 2011] Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 315–323.

[Golle, 2008] Golle, P. (2008). Machine learning attacks against the asirra captcha. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pages 535–542. Association for Computing Machinery.

[Gong et al., 2016] Gong, W., Zhang, X., Gonzàlez, J., Sobral, A., Bouwmans, T., Tu, C., and Zahzah, E.-h. (2016). Human pose estimation from monocular images : A comprehensive survey. *Sensors*, 16(12) :1966.

[Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

[Goodfellow, 2016] Goodfellow, I. (2016). Nips 2016 tutorial : Generative adversarial networks. *arXiv preprint arXiv :1701.00160*.

[Grammalidis et al., 2001] Grammalidis, N., Goussis, G., Troufakos, G., and Strintzis, M. G. (2001). 3-d human body tracking from depth images using analysis by synthesis. In *Proceedings of the IEEE International Conference on Image Processing*, volume 2, pages 185–188. IEEE.

[Grandvalet and Bengio, 2006] Grandvalet, Y. and Bengio, Y. (2006). Entropy regularization. *Semi-supervised learning*, pages 151–168.

[Graves and Jaitly, 2014] Graves, A. and Jaitly, N. (2014). Towards end-to-end speech recognition with recurrent neural networks. In *Proceedinf of the 31st International Conference on Machine Learning*, pages 1764–1772.

[Grest et al., 2005] Grest, D., Woetzel, J., and Koch, R. (2005). Nonlinear body pose estimation from depth images. In *Joint Pattern Recognition Symposium*, pages 285–292. Springer.

[Guan et al., 2009] Guan, P., Weiss, A., Balan, A. O., and Black, M. J. (2009). Estimating human shape and pose from a single image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1381–1388. IEEE.

[Gühring, 2000] Gühring, J. (2000). Dense 3d surface acquisition by structured light using off-the-shelf components. In *Videometrics and Optical Methods for 3D Shape Measurement*, volume 4309, pages 220–231. International Society for Optics and Photonics.

[Ha and Roquelaure, 2010] Ha, C. and Roquelaure, Y. (2010). Troubles musculosquelettiques d'origine professionnelle en france. ou en est-on aujourd'hui. *Bulletin épidémiologique hebdomadaire*, pages 5–6.

[Haque et al., 2016] Haque, A., Peng, B., Luo, Z., Alahi, A., Yeung, S., and Fei-Fei, L. (2016). Towards viewpoint invariant 3d human pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 160–177. Springer.

[Hartley and Zisserman, 2004] Hartley, R. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press.

[Hartzell et al., 2017] Hartzell, M. M., Dodd, C., and Gatchel, R. J. (2017). Stress and musculoskeletal injury. *The handbook of stress and health*, pages 210–222.

[Hastie et al., 2009] Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The elements of statistical learning : Data mining, inference, and prediction*. Springer Science & Business Media.

[Hatzfeld, 2006] Hatzfeld, N. (2006). L'émergence des troubles musculo-squelettiques (1982-1996). sensibilités de terrain, définitions d'experts et débats scientifiques. *Histoire & mesure*, 21(XXI-1) :111–140.

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778. IEEE.

[He et al., 2004] He, X., Zemel, R. S., and Carreira-Perpiñán, M. Á. (2004). Multiscale conditional random fields for image labeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 695–702. IEEE.

[Hebb, 1949] Hebb, D. O. (1949). *The organization of behavior : A neuropsychological theory*. J. Wiley ; Chapman & Hall.

[Heimann et al., 2004] Heimann, T., Thorn, M., Kunert, T., and Meinzer, H. (2004). New methods for leak detection and contour correction in seeded region growing segmentation. *International Archives of Photogrammetry and Remote Sensing*, 35.

[Hesse et al., 2015] Hesse, N., Stachowiak, G., Breuer, T., and Arens, M. (2015). Estimating body pose of infants in depth images using random ferns. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 35–43. IEEE.

[Hinton et al., 1999] Hinton, G. E., Sejnowski, T. J., Poggio, T. A., et al. (1999). *Unsupervised learning : Foundations of neural computation*. MIT press.

[Ho, 1995] Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282. IEEE.

[Hochreiter, 1998] Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02) :107–116.

[Hofmann and Gavrila, 2012] Hofmann, M. and Gavrila, D. M. (2012). Multi-view 3d human pose estimation in complex environment. *International Journal of Computer Vision*, 96(1) :103–124.

[Hogg, 1983] Hogg, D. (1983). Model-based vision : A program to see a walking person. *Image and Vision Computing*, 1(1) :5–20.

[Holt et al., 2011] Holt, B., Ong, E.-J., Cooper, H., and Bowden, R. (2011). Putting the pieces together : Connected poselets for human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1196–1201. IEEE.

[Horn and Schunck, 1981] Horn, B. K. and Schunck, B. G. (1981). Determining optical flow. In *Techniques and Applications of Image Understanding*, volume 281, pages 319–331. International Society for Optics and Photonics.

[Horowitz, 1975] Horowitz, S. L. (1975). Picture segmentation by a directed split-and-merge procedure. In *International Joint Conference on Pattern Recognition*, pages 424–433.

[Horowitz and Pavlidis, 1976] Horowitz, S. L. and Pavlidis, T. (1976). Picture segmentation by a tree traversal algorithm. *Journal of the ACM*, 23(2) :368–388.

[Hough, 1962] Hough, P. V. (1962). Method and means for recognizing complex patterns. US Patent 3,069,654.

[Huang et al., 2017] Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708.

[Huang et al., 2010] Huang, S.-j., Jin, R., and Zhou, Z.-h. (2010). Active learning by querying informative and representative examples. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 892–900. IEEE.

[Huang et al., 2018] Huang, X., Liu, M.-Y., Belongie, S., and Kautz, J. (2018). Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision*, pages 172–189. Springer.

[Hussmann et al., 2008] Hussmann, S., Ringbeck, T., and Hagebeuker, B. (2008). A performance review of 3d tof vision systems in comparison to stereo vision systems. In *Stereo vision*, volume 372, pages 103–120. IntechOpen.

[INRS, 2011a] INRS (2011a). Les troubles musculo-squelettiques du membre supérieur (tms-ms). Institut national de recherche et de sécurité pour la prévention des accidents du travail et des maladies professionnelles.

[INRS, 2011b] INRS (2011b). Vous avez dit tms ? Technical report, Institut national de recherche et de sécurité pour la prévention des accidents du travail et des maladies professionnelles.

[INRS, 2015] INRS (2015). Les troubles musculo-squelettiques. Institut national de recherche et de sécurité pour la prévention des accidents du travail et des maladies professionnelles.

[Insafutdinov et al., 2016] Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., and Schiele, B. (2016). Deepercut : A deeper, stronger, and faster multi-person pose estimation model. In *Proceedings of the European Conference on Computer Vision*, pages 34–50. Springer.

[Insafutdinov et al., 2017] Insafutdinov, E., Andriluka, M., Pishchulin, L., Tang, S., Levinkov, E., Andres, B., and Schiele, B. (2017). Arttrack : Articulated multi-person tracking in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6457–6465.

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch normalization : Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 448–456.

[Ionescu et al., 2013] Ionescu, C., Papava, D., Olaru, V., and Sminchisescu, C. (2013). Human3. 6m : Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7) :1325–1339.

[Iqbal and Gall, 2016] Iqbal, U. and Gall, J. (2016). Multi-person pose estimation with local joint-to-person associations. In *Proceedings of the European Conference on Computer Vision*, pages 627–642. Springer.

[ISO, 2000] ISO (2000). Ergonomie — Évaluation des postures de travail statiques. ISO 11226 :2000, Organization Internationale de Normalisation.

[ISO, 2007] ISO (2007). Ergonomie — manutention manuelle — partie 3 : Manipulation de charges faibles à fréquence de répétition élevée. ISO 11228-3 :2007, Organization Internationale de Normalisation.

[Isola et al., 2017] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1125–1134. IEEE.

[Jain et al., 2014] Jain, A., Tompson, J., Andriluka, M., Taylor, G. W., and Bregler, C. (2014). Learning human pose estimation features with convolutional networks. In *International Conference on Learning Representations*, pages 1–14.

[Jain et al., 2011] Jain, H. P., Subramanian, A., Das, S., and Mittal, A. (2011). Real-time upper-body human pose estimation using a depth camera. In *International Conference on Computer Vision/Computer Graphics Collaboration Techniques*, pages 227–238. Springer.

[Jiang, 2010] Jiang, H. (2010). 3d human pose reconstruction using millions of exemplars. In *Proceedings of the IEEE International Conference on Pattern Recognition*, pages 1674–1677. IEEE.

[Jiang et al., 2018] Jiang, Y., Krishnan, D., Mobahi, H., and Bengio, S. (2018). Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv :1810.00113*.

[Jing et al., 2019] Jing, Y., Yang, Y., Feng, Z., Ye, J., Yu, Y., and Song, M. (2019). Neural style transfer : A review. *IEEE Transactions on Visualization and Computer Graphics*.

[Jiu et al., 2014] Jiu, M., Wolf, C., Taylor, G., and Baskurt, A. (2014). Human body part estimation from depth images via spatially-constrained deep learning. *Pattern Recognition Letters*, 50 :122–129.

[Joachims, 1999] Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning*, volume 99, pages 200–209. Association for Computing Machinery.

[Johnson et al., 2016] Johnson, J., Alahi, A., and Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision*, pages 694–711. Springer.

[Johnson and Everingham, 2010] Johnson, S. and Everingham, M. (2010). Clustered pose and nonlinear appearance models for human pose estimation. In *British Machine Vision Conference*, volume 2, page 5. British Machine Vision Association.

[Johnson and Everingham, 2011] Johnson, S. and Everingham, M. (2011). Learning effective human pose estimation from inaccurate annotation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1465–1472. IEEE.

[Jordan and Jacobs, 1994] Jordan, M. I. and Jacobs, R. A. (1994). Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6(2) :181–214.

[Ju et al., 1996] Ju, S. X., Black, M. J., and Yacoob, Y. (1996). Cardboard people : A parameterized model of articulated image motion. In *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, pages 38–44. IEEE.

[Julesz, 1981] Julesz, B. (1981). Textons, the elements of texture perception, and their interactions. *Nature*, 290(5802) :91–97.

[Kanaujia et al., 2007] Kanaujia, A., Sminchisescu, C., and Metaxas, D. (2007). Semi-supervised hierarchical models for 3d human pose reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Kass et al., 1987] Kass, M., Witkin, A., and Terzopoulos, D. (1987). Snakes : Active contour models. *International Journal of Computer Vision*, 1(4) :321–331.

[Kendall et al., 2017] Kendall, A., Badrinarayanan, V., and Cipolla, R. (2017). Bayesian segnet : Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. In *British Machine Vision Conference*. British Machine Vision Association.

[Kim et al., 2017] Kim, T., Cha, M., Kim, H., Lee, J. K., and Kim, J. (2017). Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 1857–1865.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. (2015). Adam : A method for stochastic optimization. In *International Conference on Learning Representations*.

[Knoop et al., 2006] Knoop, S., Vacek, S., and Dillmann, R. (2006). Sensor fusion for 3d human body tracking with an articulated 3d body model. In *IEEE International Conference on Robotics and Automation*, pages 1686–1691. IEEE.

[Kocabas et al., 2018] Kocabas, M., Karagoz, S., and Akbas, E. (2018). Multiposenet : Fast multi-person pose estimation using pose residual network. In *Proceedings of the European Conference on Computer Vision*, pages 417–433.

[Kramer, 1991] Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE journal*, 37 :233–243.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105.

[Krogel and Scheffer, 2004] Krogel, M.-A. and Scheffer, T. (2004). Multi-relational learning, text mining, and semi-supervised learning for functional genomics. *Machine Learning*, 57 :61–81.

[Lachat et al., 2015] Lachat, E., Macher, H., Landes, T., and Grussenmeyer, P. (2015). Assessment and calibration of a rgb-d camera (kinect v2 sensor) towards a potential use for close-range 3d modeling. *Remote Sensing*, 7(10) :13070–13097.

[Lafferty et al., 2001] Lafferty, J., McCallum, A., and Pereira, F. C. . (2001). Conditional random fields : Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289. Association for Computing Machinery.

[Lan and Huttenlocher, 2005] Lan, X. and Huttenlocher, D. P. (2005). Beyond trees : Common-factor models for 2d human pose recovery. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 1, pages 470–477. IEEE.

[Lazebnik et al., 2006] Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features : Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2169–2178. IEEE.

[Le Cun, 1986] Le Cun, Y. (1986). Learning process in an asymmetric threshold network. In *Disordered Systems and Biological Organization*, pages 233–240. Springer.

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11) :2278–2324.

[Lee and Cohen, 2004] Lee, M. W. and Cohen, I. (2004). Human upper body pose estimation in static images. In *Proceedings of the European Conference on Computer Vision*, pages 126–138. Springer.

[Leibe et al., 2008] Leibe, B., Leonardis, A., and Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, 77(1-3) :259–289.

[Lewis and Gale, 1994] Lewis, D. D. and Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12. Springer.

[Li et al., 2019] Li, R., Liu, Z., and Tan, J. (2019). A survey on 3d hand pose estimation : Cameras, methods, and datasets. *Pattern Recognition*, 93 :251–272.

[Li and Chan, 2014] Li, S. and Chan, A. B. (2014). 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision*, pages 332–347. Springer.

[Li et al., 2015] Li, S., Zhang, W., and Chan, A. B. (2015). Maximum-margin structured learning with deep networks for 3d human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2848–2856.

[Li et al., 2010] Li, W., Zhang, Z., and Liu, Z. (2010). Action recognition based on a bag of 3d points. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 9–14. IEEE.

[Liang et al., 2018] Liang, X., Gong, K., Shen, X., and Lin, L. (2018). Look into person : Joint body parsing & pose estimation network and a new benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4) :871–885.

[Lin et al., 2017] Lin, G., Milan, A., Shen, C., and Reid, I. (2017). Refinenet : Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1925–1934.

[Lindner et al., 2010] Lindner, M., Schiller, I., Kolb, A., and Koch, R. (2010). Time-of-flight sensor calibration for accurate range sensing. *Computer Vision and Image Understanding*, 114(12) :1318–1328.

[Liu and Tuzel, 2016] Liu, M.-Y. and Tuzel, O. (2016). Coupled generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 469–477.

[Liu et al., 2017] Liu, M.-Y., Breuel, T., and Kautz, J. (2017). Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*, pages 700–708.

[Liu et al., 2019] Liu, M.-Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., and Kautz, J. (2019). Few-shot unsupervised image-to-image translation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 10551–10560.

[Liu et al., 2015a] Liu, W., Rabinovich, A., and Berg, A. C. (2015a). Parsenet : Looking wider to see better. *arXiv preprint arXiv :1506.04579*.

[Liu et al., 2015b] Liu, Z., Zhu, J., Bu, J., and Chen, C. (2015b). A survey of human pose estimation : the body parts parsing based methods. *Journal of Visual Communication and Image Representation*, 32 :10–19.

[Lonarkar and Rao, 2017] Lonarkar, V. and Rao, B. A. (2017). Content-based image retrieval by segmentation and clustering. In *International Conference on Inventive Computing and Informatics*, pages 771–776. IEEE.

[Long et al., 2015] Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440. IEEE.

[Lowe, 1999] Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157. IEEE.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2) :91–110.

[Luvizon et al., 2018] Luvizon, D. C., Picard, D., and Tabia, H. (2018). 2d/3d pose estimation and action recognition using multitask deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5137–5146.

[Luvizon et al., 2019] Luvizon, D. C., Tabia, H., and Picard, D. (2019). Human pose regression by combining indirect part detection and contextual information. *Computers & Graphics*, 85 :15–22.

[Ma et al., 2017] Ma, J., Jiang, J., Liu, C., and Li, Y. (2017). Feature guided gaussian mixture model with semi-supervised em and local geometric constraint for retinal image registration. *Information Sciences*, 417 :128–142.

[Maas et al., 2013] Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning Workshop on Deep Learning for Audio, Speech and Language Processing.*

[Maimone and Fuchs, 2012] Maimone, A. and Fuchs, H. (2012). Reducing interference between multiple structured light depth sensors using motion. In *Proceedings of the IEEE Virtual Reality Workshops*, pages 51–54. IEEE.

[Malakoff Médéric, 2015] Malakoff Médéric (2015). Facteurs de risque des tms. https://entreprise-territoire-de-sante.malakoffmederic.com/risques-professionnels/info/les-facteurs-de-risque-des-tms#les-facteurs-de-risque-individuels.

[Mao et al., 2016] Mao, X., Li, Q., Xie, H., Lau, R. Y., and Wang, Z. (2016). Multi-class generative adversarial networks with the l2 loss function. *arXiv preprint arXiv :1611.04076*, 5.

[Marín-Jiménez et al., 2018] Marín-Jiménez, M. J., Romero-Ramirez, F. J., Muñoz-Salinas, R., and Medina-Carnicer, R. (2018). 3d human pose estimation from depth maps using a deep combination of poses. *Journal of Visual Communication and Image Representation*, 55 :627–639.

[Masulli and Schenone, 1999] Masulli, F. and Schenone, A. (1999). A fuzzy clustering based segmentation system as support to diagnosis in medical imaging. *Artificial Intelligence in Medicine*, 16(2) :129–147.

[McCulloch and Pitts, 1943] McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4) :115–133.

[McGuinness and O'connor, 2010] McGuinness, K. and O'connor, N. (2010). A comparative evaluation of interactive segmentation algorithms. *Pattern Recognition*, 43(2) :434–444.

[Medina-Carnicer et al., 2011] Medina-Carnicer, R., Munoz-Salinas, R., Yeguas-Bolivar, E., and Diaz-Mas, L. (2011). A novel method to look for the hysteresis thresholds for the canny edge detector. *Pattern Recognition*, 44(6) :1201–1211.

[Mehta et al., 2017a] Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., and Theobalt, C. (2017a). Monocular 3d human pose estimation in the wild using improved cnn supervision. In *5th International Conference on 3D Vision*, pages 506–516. IEEE.

[Mehta et al., 2017b] Mehta, D., Sridhar, S., Sotnychenko, O., Rhodin, H., Shafiei, M., Seidel, H.-P., Xu, W., Casas, D., and Theobalt, C. (2017b). Vnect : Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics*, 36(4) :1–14.

[Mehta et al., 2018] Mehta, D., Sotnychenko, O., Mueller, F., Xu, W., Sridhar, S., Pons-Moll, G., and Theobalt, C. (2018). Single-shot multi-person 3d pose estimation from monocular rgb. In *6th International Conference on 3D Vision*, pages 120–130. IEEE.

[Mejbri et al., 2019] Mejbri, S., Franchet, C., Reshma, I. A., Mothe, J., Brousset, P., and Faure, E. (2019). Deep analysis of cnn settings for new cancer whole-slide histological images segmentation : The case of small training sets. In *Proceedings of the 12th International Joint Conference on Biomedical Engineering Systems and Technology*, volume 2, pages 120–128.

[Ménier et al., 2006] Ménier, C., Boyer, E., and Raffin, B. (2006). 3d skeleton-based body pose recovery. In *3rd International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 389–396. IEEE.

[Ministère de la santé, 2012] Ministère de la santé (2012). Tableau des maladies professionnelles numéro 57. Tableau.

[Ministère de la santé, 2020] Ministère de la santé (2020). Annexe II : Tableaux des maladies professionnelles prévus à l'article R. 461-3. Tableau.

[Ministère du travail, 2014] Ministère du travail (2014). Conditions de travail - bilan 2014. http://travail-emploi.gouv.fr/ministere/documentation-et-publications-officielles/rapports/article/conditions-de-travail-bilan-2014.

[Minsky and Papert, 1969] Minsky, M. L. and Papert, S. (1969). *Perceptrons : An Introduction to Computational Geometry*. MIT Press.

[Mirza and Osindero, 2014] Mirza, M. and Osindero, S. (2014). Conditional generative adversarial nets. *arXiv preprint arXiv :1411.1784*.

[Moon et al., 2018] Moon, G., Yong Chang, J., and Mu Lee, K. (2018). V2v-posenet : Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5079–5088. IEEE.

[Moreno-Noguer, 2017] Moreno-Noguer, F. (2017). 3d human pose estimation from a single image via distance matrix regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2823–2832.

[Mori et al., 2004] Mori, G., Ren, X., Efros, A. A., and Malik, J. (2004). Recovering human body configurations : Combining segmentation and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 326–333. IEEE.

[Mori et al., 2005] Mori, G., Belongie, S., and Malik, J. (2005). Efficient shape matching using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(11) :1832–1837.

[Mori and Malik, 2006] Mori, G. and Malik, J. (2006). Recovering 3d human body configurations using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7) :1052–1062.

[Mori et al., 2015] Mori, G., Pantofaru, C., Kothari, N., Leung, T., Toderici, G., Toshev, A., and Yang, W. (2015). Pose embeddings : A deep architecture for learning to match human poses. *arXiv preprint arXiv :1507.00302*.

[Mumford and Shah, 1989] Mumford, D. and Shah, J. (1989). Optimal approximations by piecewise smooth functions and associated variational problems. *Communications on Pure and Applied Mathematics*, 42(5) :577–685.

[Newell et al., 2016] Newell, A., Yang, K., and Deng, J. (2016). Stacked hourglass networks for human pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 483–499. Springer.

[Newell et al., 2017] Newell, A., Huang, Z., and Deng, J. (2017). Associative embedding : End-to-end learning for joint detection and grouping. In *Advances in Neural Information Processing Systems*, pages 2277–2287.

[Nguyen and Smeulders, 2004] Nguyen, H. T. and Smeulders, A. (2004). Active learning using pre-clustering. In *Proceedings of the 21st International Conference on Machine Learning*, pages 79–87. Association for Computing Machinery.

[Nickel and Stiefelhagen, 2007] Nickel, K. and Stiefelhagen, R. (2007). Visual recognition of pointing gestures for human-robot interaction. *Image and Vision Computing*, 25(12) :1875–1884.

[Nie et al., 2017] Nie, B. X., Wei, P., and Zhu, S.-C. (2017). Monocular 3d human pose estimation by predicting depth on joints. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3467–3475. IEEE.

[Nie et al., 2018] Nie, X., Feng, J., Zuo, Y., and Yan, S. (2018). Human pose estimation with parsing induced learner. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2100–2108.

[Ning et al., 2017] Ning, G., Zhang, Z., and He, Z. (2017). Knowledge-guided deep fractal neural networks for human pose estimation. *Transactions on Multimedia*, 20(5) :1246–1259.

[Ning et al., 2008] Ning, H., Xu, W., Gong, Y., and Huang, T. (2008). Discriminative learning of visual words for 3d human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Ning et al., 2010] Ning, J., Zhang, L., Zhang, D., and Wu, C. (2010). Interactive image segmentation by maximal similarity based region merging. *Pattern Recognition*, 43(2) :445–456.

[Nister and Stewenius, 2006] Nister, D. and Stewenius, H. (2006). Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 2161–2168. IEEE.

[Obdržálek et al., 2012] Obdržálek, Š., Kurillo, G., Ofli, F., Bajcsy, R., Seto, E., Jimison, H., and Pavel, M. (2012). Accuracy and robustness of kinect pose estimation in the context of coaching of elderly population. In *Proceedings of the IEEE International Conference on Engineering in Medicine and Biology Society*, pages 1188–1193. IEEE.

[Odena et al., 2017] Odena, A., Olah, C., and Shlens, J. (2017). Conditional image synthesis with auxiliary classifier gans. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2642–2651.

[Ohlander et al., 1978] Ohlander, R., Price, K., and Reddy, D. R. (1978). Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8(3) :313–333.

[Okada and Soatto, 2008] Okada, R. and Soatto, S. (2008). Relevant feature selection for human pose estimation and localization in cluttered images. In *Proceedings of the European Conference on Computer Vision*, pages 434–445. Springer.

[Otsu, 1979] Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1) :62–66.

[Ott and Everingham, 2009] Ott, P. and Everingham, M. (2009). Implicit color segmentation features for pedestrian and object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 723–730. IEEE.

[Pan and Yang, 2009] Pan, S. J. and Yang, Q. (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10) :1345–1359.

[Pan et al., 2010] Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q. (2010). Domain adaptation via transfer component analysis. *Transactions on Neural Networks*, 22(2) :199–210.

[Papandreou et al., 2017] Papandreou, G., Zhu, T., Kanazawa, N., Toshev, A., Tompson, J., Bregler, C., and Murphy, K. (2017). Towards accurate multi-person pose estimation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4903–4911. IEEE.

[Papandreou et al., 2018] Papandreou, G., Zhu, T., Chen, L.-C., Gidaris, S., Tompson, J., and Murphy, K. (2018). Personlab : Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *Proceedings of the European Conference on Computer Vision*, pages 269–286.

[Paszke et al., 2016] Paszke, A., Chaurasia, A., Kim, S., and Culurciello, E. (2016). Enet : A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv :1606.02147*.

[Pavlakos et al., 2017] Pavlakos, G., Zhou, X., Derpanis, K. G., and Daniilidis, K. (2017). Coarse-to-fine volumetric prediction for single-image 3d human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7025–7034.

[Pavlakos et al., 2018] Pavlakos, G., Zhou, X., and Daniilidis, K. (2018). Ordinal depth supervision for 3d human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7307–7316.

[Pearson, 1901] Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2 :559–572.

[Peng et al., 2011] Peng, B., Zhang, L., and Zhang, D. (2011). Automatic image segmentation by dynamic region merging. *IEEE Transactions on Image Processing*, 20(12) :3592–3605.

[Perarnau et al., 2016] Perarnau, G., Van De Weijer, J., Raducanu, B., and Álvarez, J. M. (2016). Invertible conditional gans for image editing. In *Advances in Neural Information Processing Systems Workshop*.

[Perez-Sala et al., 2014] Perez-Sala, X., Escalera, S., Angulo, C., and Gonzalez, J. (2014). A survey on model based approaches for 2d and 3d visual human pose recovery. *Sensors*, 14(3) :4189–4210.

[Pfister et al., 2015] Pfister, T., Charles, J., and Zisserman, A. (2015). Flowing convnets for human pose estimation in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1913–1921. IEEE.

[Pham et al., 2000] Pham, D. L., Xu, C., and Prince, J. L. (2000). Current methods in medical image segmentation. *Annual Review of Biomedical Engineering*, 2(1) :315–337.

[Pinheiro and Collobert, 2014] Pinheiro, P. O. and Collobert, R. (2014). Recurrent convolutional neural networks for scene labeling. In *Proceedings of the 31st International Conference on Machine Learning*.

[Pinheiro et al., 2015] Pinheiro, P. O., Collobert, R., and Dollár, P. (2015). Learning to segment object candidates. In *Advances in Neural Information Processing Systems*, pages 1990–1998.

[Pinheiro et al., 2016] Pinheiro, P. O., Lin, T.-Y., Collobert, R., and Dollár, P. (2016). Learning to refine object segments. In *Proceedings of the European Conference on Computer Vision*, pages 75–91. Springer.

[Pishchulin et al., 2013] Pishchulin, L., Andriluka, M., Gehler, P., and Schiele, B. (2013). Poselet conditioned pictorial structures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–595.

[Pishchulin et al., 2016] Pishchulin, L., Insafutdinov, E., Tang, S., Andres, B., Andriluka, M., Gehler, P. V., and Schiele, B. (2016). Deepcut : Joint subset partition and labeling for multi person pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4929–4937. IEEE.

[Plagemann et al., 2010] Plagemann, C., Ganapathi, V., Koller, D., and Thrun, S. (2010). Real-time identification and localization of body parts from depth images. In *IEEE International Conference on Robotics and Automation*, pages 3108–3113. IEEE.

[Plantard et al., 2015] Plantard, P., Auvinet, E., Pierres, A.-S. L., and Multon, F. (2015). Pose estimation with a kinect for ergonomic studies : Evaluation of the accuracy using a virtual mannequin. *Sensors*, 15(1) :1785–1803.

[Plantard et al., 2017a] Plantard, P., Muller, A., Pontonnier, C., Dumont, G., Shum, H. P., and Multon, F. (2017a). Inverse dynamics based on occlusion-resistant kinect data : Is it usable for ergonomics ? *International Journal of Industrial Ergonomics*, 61 :71–80.

[Plantard et al., 2017b] Plantard, P., Shum, H. P., Le Pierres, A.-S., and Multon, F. (2017b). Validation of an ergonomic assessment method using kinect data in real workplace conditions. *Applied ergonomics*, 65 :562–569.

[Pock et al., 2009] Pock, T., Cremers, D., Bischof, H., and Chambolle, A. (2009). An algorithm for minimizing the mumford-shah functional. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1133–1140. IEEE.

[Pohle and Toennies, 2001] Pohle, R. and Toennies, K. D. (2001). Segmentation of medical images using adaptive region growing. In *Medical Imaging 2001 : Image Processing*, volume 4322, pages 1337–1346. International Society for Optics and Photonics.

[Poppe and Poel, 2006] Poppe, R. and Poel, M. (2006). Comparison of silhouette shape descriptors for example-based human pose recovery. In *Proceedings of the 7th International Conference on Automatic Face and Gesture Recognition*, pages 541–546. IEEE.

[Poppe, 2007a] Poppe, R. (2007a). Evaluating example-based pose estimation : Experiments on the humaneva sets. *Applied Mathematics Letters*.

[Poppe, 2007b] Poppe, R. (2007b). Vision-based human motion analysis : An overview. *Computer Vision and Image Understanding*, 108(1-2) :4–18.

[Prewitt, 1970] Prewitt, J. M. (1970). Object enhancement and extraction. *Picture Processing and Psychopictorics*, 10(1) :15–19.

[Pujol et al., 2017] Pujol, F. A., Pujol, M., Jimeno-Morenilla, A., and Pujol, M. J. (2017). Face detection based on skin color segmentation using fuzzy entropy. *Entropy*, 19(1) :26.

[Punnett and Wegman, 2004] Punnett, L. and Wegman, D. H. (2004). Work-related musculoskeletal disorders : the epidemiologic evidence and the debate. *Journal of electromyography and kinesiology*, 14(1) :13–23.

[Qi et al., 2017a] Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017a). Pointnet++ : Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108.

[Qi et al., 2017b] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017b). Pointnet : Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660. IEEE.

[Qian and Huang, 1996] Qian, R. J. and Huang, T. S. (1996). Optimal edge detection in two-dimensional images. *IEEE Transactions on Image Processing*, 5(7) :1215–1220.

[Radford et al., 2015] Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*.

[Raina et al., 2007] Raina, R., Battle, A., Lee, H., Packer, B., and Ng, A. Y. (2007). Self-taught learning : transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning*, pages 759–766. Association for Computing Machinery.

[Ramakrishna et al., 2014] Ramakrishna, V., Munoz, D., Hebert, M., Bagnell, J. A., and Sheikh, Y. (2014). Pose machines : Articulated pose estimation via inference machines. In *Proceedings of the European Conference on Computer Vision*, pages 33–47. Springer.

[Ranzato et al., 2007] Ranzato, M., Huang, F. J., Boureau, Y.-L., and LeCun, Y. (2007). Unsupervised learning of invariant feature hierarchies with applications to object re-

cognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Ray and Turi, 1999] Ray, S. and Turi, R. H. (1999). Determination of number of clusters in k-means clustering and application in colour image segmentation. In *Proceedings of the 4th International Conference on Advances in Pattern Recognition and Digital Techniques*, pages 137–143.

[Ray and Teizer, 2012] Ray, S. J. and Teizer, J. (2012). Real-time construction worker posture analysis for ergonomics training. *Advanced Engineering Informatics*, 26(2) :439–455.

[Reed et al., 2016] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative adversarial text to image synthesis. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 1060–1069.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn : Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99.

[Rogez and Schmid, 2016] Rogez, G. and Schmid, C. (2016). Mocap-guided data augmentation for 3d pose estimation in the wild. In *Advances in Neural Information Processing Systems*, pages 3108–3116.

[Rogez et al., 2017] Rogez, G., Weinzaepfel, P., and Schmid, C. (2017). Lcr-net : Localization-classification-regression for human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3433–3441.

[Rogez et al., 2019] Rogez, G., Weinzaepfel, P., and Schmid, C. (2019). Lcr-net++ : Multi-person 2d and 3d pose detection in natural images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net : Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer.

[Rosales and Sclaroff, 2000] Rosales, R. and Sclaroff, S. (2000). Inferring body pose without tracking body parts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 721–727. IEEE.

[Rosales and Sclaroff, 2002] Rosales, R. and Sclaroff, S. (2002). Learning body pose via specialized maps. In *Advances in Neural Information Processing Systems*, pages 1263–1270.

[Rosenberg et al., 2005] Rosenberg, C., Hebert, M., and Schneiderman, H. (2005). Semi-supervised self-training of object detection models. In *7th IEEE Workshops on Application of Computer Vision*, volume 1, pages 29–36.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron : A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6) :386.

[Roy and Todorovic, 2016] Roy, A. and Todorovic, S. (2016). A multi-scale cnn for affordance segmentation in rgb images. In *Proceedings of the European Conference on Computer Vision*, pages 186–201. Springer.

[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088) :533–536.

[Russell and Norvig, 2002] Russell, S. and Norvig, P. (2002). *Artificial Intelligence : A Modern Approach (International Edition)*. Pearson Education.

[Réseau Anact, 2008] Réseau Anact (2008). L'approche économique des tms... Brochure.

[Salimans et al., 2016] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242.

[Sangkloy et al., 2017] Sangkloy, P., Lu, J., Fang, C., Yu, F., and Hays, J. (2017). Scribbler : Controlling deep image synthesis with sketch and color. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5400–5409. IEEE.

[Sapp et al., 2010a] Sapp, B., Jordan, C., and Taskar, B. (2010a). Adaptive pose priors for pictorial structures. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 422–429. IEEE.

[Sapp et al., 2010b] Sapp, B., Toshev, A., and Taskar, B. (2010b). Cascaded models for articulated pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 406–420. Springer.

[Sapp and Taskar, 2013] Sapp, B. and Taskar, B. (2013). Modec : Multimodal decomposable models for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3681.

[Schwarz et al., 2011] Schwarz, L. A., Mkhitaryan, A., Mateus, D., and Navab, N. (2011). Estimating human 3d pose from time-of-flight images based on geodesic distances and optical flow. In *Proceedings of the 9th International Conference on Automatic Face an Gesture Recognition*, pages 700–706. IEEE.

[Scovanner et al., 2007] Scovanner, P., Ali, S., and Shah, M. (2007). A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM International Conference on Multimedia*, pages 357–360. Association for Computing Machinery.

[Serre et al., 2005] Serre, T., Wolf, L., and Poggio, T. (2005). Object recognition with features inspired by visual cortex. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 994–1000. IEEE.

[Seung et al., 1992] Seung, H. S., Opper, M., and Sompolinsky, H. (1992). Query by committee. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pages 287–294.

[Shahroudy et al., 2016] Shahroudy, A., Liu, J., Ng, T.-T., and Wang, G. (2016). Ntu rgb+d : A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

[Shakhnarovich et al., 2003] Shakhnarovich, G., Viola, P., and Darrell, T. (2003). Fast pose estimation with parameter-sensitive hashing. In *Proceedings of the IEEE International Conference on Computer Vision*, page 750. IEEE.

[Shi and Malik, 2000] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8) :888–905.

[Shotton et al., 2006] Shotton, J., Winn, J., Rother, C., and Criminisi, A. (2006). Textonboost : Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 1–15. Springer.

[Shotton et al., 2008] Shotton, J., Johnson, M., and Cipolla, R. (2008). Semantic texton forests for image categorization and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE.

[Shotton et al., 2011] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., and Blake, A. (2011). Real-time human pose recognition in parts from single depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1297–1304. IEEE.

[Shrivastava et al., 2017] Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. (2017). Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2107–2116. IEEE.

[Shum et al., 2013] Shum, H. P., Ho, E. S., Jiang, Y., and Takagi, S. (2013). Real-time posture reconstruction for microsoft kinect. *IEEE Transactions on Cybernetics*, 43(5) :1357–1369.

[Shvets et al., 2018] Shvets, A. A., Rakhlin, A., Kalinin, A. A., and Iglovikov, V. I. (2018). Automatic instrument segmentation in robot-assisted surgery using deep learning. In *17th IEEE International Conference on Machine Learning and Applications*, pages 624–628. IEEE.

[Siddiqui and Medioni, 2010] Siddiqui, M. and Medioni, G. (2010). Human pose estimation from a single view point, real-time range sensor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE.

[Sidenbladh et al., 2000] Sidenbladh, H., De la Torre, F., and Black, M. J. (2000). A framework for modeling the appearance of 3d articulated figures. In *Proceedings of the*

4th IEEE International Conference on Automatic Face and Gesture Recognition, pages 368–375. IEEE.

[Sigal et al., 2004] Sigal, L., Bhatia, S., Roth, S., Black, M. J., and Isard, M. (2004). Tracking loose-limbed people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 421–428. IEEE.

[Sigal and Black, 2006] Sigal, L. and Black, M. J. (2006). Predicting 3d people from 2d pictures. In *International Conference on Articulated Motion and Deformable Objects*, pages 185–195. Springer.

[Sigal et al., 2010] Sigal, L., Balan, A. O., and Black, M. J. (2010). Humaneva : Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87(1-2) :4.

[Sigal et al., 2012] Sigal, L., Isard, M., Haussecker, H., and Black, M. J. (2012). Loose-limbed people : Estimating 3d human pose and motion using non-parametric belief propagation. *International Journal of Computer Vision*, 98(1) :15–48.

[Silver et al., 2016] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587) :484.

[Simoneau et al., 2012] Simoneau, S., St-Vincent, M., and Chicoine, D. (2012). Les tms des membres supérieurs : mieux les comprendre pour mieux les prévenir. *IRSST*.

[Simonyan and Zisserman, 2015] Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*.

[Sminchisescu and Triggs, 2001] Sminchisescu, C. and Triggs, B. (2001). Covariance scaled sampling for monocular 3d body tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 447–454. IEEE.

[Sminchisescu and Telea, 2002] Sminchisescu, C. and Telea, A. (2002). Human pose estimation from silhouettes : A consistent approach using distance level sets. In *10th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, volume 10, pages 413–420.

[Sminchisescu and Triggs, 2003] Sminchisescu, C. and Triggs, B. (2003). Estimating articulated human motion with covariance scaled sampling. *International Journal of Robotics Research*, 22(6) :371–391.

[Sminchisescu and Jepson, 2004] Sminchisescu, C. and Jepson, A. (2004). Generative modeling for continuous non-linearly embedded visual inference. In *Proceedings of the 21st International Conference on Machine Learning*.

[Sminchisescu et al., 2005] Sminchisescu, C., Kanaujia, A., Li, Z., and Metaxas, D. (2005). Discriminative density propagation for 3d human motion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 390–397. IEEE.

[Sobel and Feldman, 1968] Sobel, I. and Feldman, G. (1968). A 3x3 isotropic gradient operator for image processing. *A talk at the Stanford Artificial Project*, pages 271–272.

[Soler et al., 2001] Soler, L., Delingette, H., Malandain, G., Montagnat, J., Ayache, N., Koehl, C., Dourthe, O., Malassagne, B., Smith, M., Mutter, D., et al. (2001). Fully automatic anatomical, pathological, and functional segmentation from ct scans for hepatic surgery. *Computer Aided Surgery*, 6(3) :131–142.

[Steinhaus, 1956] Steinhaus, H. (1956). Sur la division des corps materiels en parties. *Bulletin de l'Académie Polonaise des Sciences*, 1(804) :801.

[Stiefelhagen et al., 2004] Stiefelhagen, R., Fugen, C., Gieselmann, R., Holzapfel, H., Nickel, K., and Waibel, A. (2004). Natural human-robot interaction using speech, head pose and gestures. In *International Conference on Intelligent Robots and Systems*, volume 3, pages 2422–2427. IEEE.

[Strekalovskiy and Cremers, 2014] Strekalovskiy, E. and Cremers, D. (2014). Real-time minimization of the piecewise smooth mumford-shah functional. In *Proceedings of the European Conference on Computer Vision*, pages 127–141. Springer.

[Sumengen and Manjunath, 2005] Sumengen, B. and Manjunath, B. (2005). Multi-scale edge detection and image segmentation. In *13th European Signal Processing Conference*, pages 1–4. IEEE.

[Sun et al., 2019] Sun, K., Xiao, B., Liu, D., and Wang, J. (2019). Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5693–5703.

[Sun et al., 2012] Sun, M., Kohli, P., and Shotton, J. (2012). Conditional regression forests for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3394–3401. IEEE.

[Sun et al., 2017] Sun, X., Shang, J., Liang, S., and Wei, Y. (2017). Compositional human pose regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2602–2611.

[Sun et al., 2018] Sun, X., Xiao, B., Wei, F., Liang, S., and Wei, Y. (2018). Integral human pose regression. In *Proceedings of the European Conference on Computer Vision*, pages 529–545.

[Suzuki and Abe, 1985] Suzuki, S. and Abe, K. (1985). Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1) :32–46.

[Szegedy et al., 2017] Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *31st AAAI Conference on Artificial Intelligence*.

[Taigman et al., 2016] Taigman, Y., Polyak, A., and Wolf, L. (2016). Unsupervised cross-domain image generation. In *International Conference on Learning Representations*.

[Tang et al., 2018] Tang, W., Yu, P., and Wu, Y. (2018). Deeply learned compositional models for human pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 190–206.

[Tekin et al., 2016] Tekin, B., Katircioglu, I., Salzmann, M., Lepetit, V., and Fua, P. (2016). Structured prediction of 3d human pose with deep neural networks. In *British Machine Vision Conference*. British Machine Vision Association.

[Tekin et al., 2017] Tekin, B., Márquez-Neila, P., Salzmann, M., and Fua, P. (2017). Learning to fuse 2d and 3d image cues for monocular body pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3941–3950.

[The MakeHuman Team, 2020] The MakeHuman Team (2020). *MakeHuman - Open Source tool for making 3D characters*.

[Tipping, 2001] Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1 :211–244.

[Tome et al., 2017] Tome, D., Russell, C., and Agapito, L. (2017). Lifting from the deep : Convolutional 3d pose estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2500–2509. IEEE.

[Tompson et al., 2014] Tompson, J. J., Jain, A., LeCun, Y., and Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems*, pages 1799–1807.

[Toshev and Szegedy, 2014] Toshev, A. and Szegedy, C. (2014). Deeppose : Human pose estimation via deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1653–1660. IEEE.

[Triguero et al., 2015] Triguero, I., García, S., and Herrera, F. (2015). Self-labeled techniques for semi-supervised learning : Taxonomy, software and empirical study. *Knowledge and Information Systems*, 42 :245–284.

[Tsai et al., 2001] Tsai, A., Yezzi, A., and Willsky, A. S. (2001). Curve evolution implementation of the mumford-shah functional for image segmentation, denoising, interpolation, and magnification. *IEEE Transactions on Image Processing*, 10(8) :1169–1186.

[Tuia et al., 2011] Tuia, D., Volpi, M., Copa, L., Kanevski, M., and Munoz-Mari, J. (2011). A survey of active learning algorithms for supervised remote sensing image classification. *Journal of Selected Topics in Signal Processing*, 5 :606–617.

[Tulyakov et al., 2018] Tulyakov, S., Liu, M.-Y., Yang, X., and Kautz, J. (2018). Mocogan : Decomposing motion and content for video generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE.

[UE, 2016] UE (2016). Règlement du parlement européen et du conseil relatif à la protection des personnes physiques à l'égard du traitement des données à caractère personnel et à la libre circulation de ces données, et abrogeant la directive 95/46/ce. Union Européenne.

[Ulyanov et al., 2016] Ulyanov, D., Vedaldi, A., and Lempitsky, V. (2016). Instance normalization : The missing ingredient for fast stylization. *arXiv preprint arXiv :1607.08022*.

[Van Laarhoven and Aarts, 1987] Van Laarhoven, P. J. and Aarts, E. H. (1987). Simulated annealing. In *Simulated annealing : Theory and applications*, pages 7–15. Springer.

[Vapnik, 1995] Vapnik, V. (1995). *The nature of statistical learning theory*. Springer Science & Business Media.

[Vedaldi and Soatto, 2008] Vedaldi, A. and Soatto, S. (2008). Quick shift and kernel methods for mode seeking. In *Proceedings of the European Conference on Computer Vision*, pages 705–718. Springer.

[Vezhnevets and Konouchine, 2005] Vezhnevets, V. and Konouchine, V. (2005). Growcut : Interactive multi-label nd image segmentation by cellular automata. In *Proceedings of Graphicon*, volume 1, pages 150–156. Citeseer.

[Vinyals et al., 2019] Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782) :350–354.

[Viola and Jones, 2001] Viola, P. and Jones, M. (2001). Robust real-time object detection. *International Journal of Computer Vision*, 4 :34–47.

[Viola et al., 2005] Viola, P., Jones, M. J., and Snow, D. (2005). Detecting pedestrians using patterns of motion and appearance. *International Journal of Computer Vision*, 63(2) :153–161.

[Visin et al., 2015] Visin, F., Kastner, K., Cho, K., Matteucci, M., Courville, A., and Bengio, Y. (2015). Renet : A recurrent neural network based alternative to convolutional networks. *arXiv preprint arXiv :1505.00393*.

[Visin et al., 2016] Visin, F., Ciccone, M., Romero, A., Kastner, K., Cho, K., Bengio, Y., Matteucci, M., and Courville, A. (2016). Reseg : A recurrent neural network-based model for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 41–48.

[Vondrick et al., 2016] Vondrick, C., Pirsiavash, H., and Torralba, A. (2016). Generating videos with scene dynamics. In *Advances in Neural Information Processing Systems*, pages 613–621.

[Wachs et al., 2011] Wachs, J. P., Kölsch, M., Stern, H., and Edan, Y. (2011). Vision-based hand-gesture applications. *Communications of the ACM*, 54(2) :60–71.

[Wang et al., 2013] Wang, C., Wang, Y., and Yuille, A. L. (2013). An approach to pose-based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 915–922. IEEE.

[Wang et al., 2014] Wang, C., Wang, Y., Lin, Z., Yuille, A. L., and Gao, W. (2014). Robust estimation of 3d human poses from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2361–2368.

[Wang et al., 2018a] Wang, C., Xu, C., Wang, C., and Tao, D. (2018a). Perceptual adversarial networks for image-to-image transformation. *IEEE Transactions on Image Processing*, 27(8) :4066–4079.

[Wang et al., 2016] Wang, K., Zhai, S., Cheng, H., Liang, X., and Lin, L. (2016). Human pose estimation from depth images via inference embedded multi-task learning. In *Proceedings of the 24th ACM International Conference on Multimedia*, pages 1227–1236. Association for Computing Machinery.

[Wang et al., 2015] Wang, Q., Kurillo, G., Ofli, F., and Bajcsy, R. (2015). Evaluation of pose tracking accuracy in the first and second generations of microsoft kinect. In *International Conference on Healthcare Informatics*, pages 380–389. IEEE.

[Wang et al., 2019a] Wang, R., Cao, Z., Wang, X., Liu, Z., and Zhu, X. (2019a). Human pose estimation with deeply learned multi-scale compositional models. *IEEE Access*, 7 :71158–71166.

[Wang et al., 2018b] Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Tao, A., Kautz, J., and Catanzaro, B. (2018b). High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8798–8807. IEEE.

[Wang et al., 2018c] Wang, T.-C., Liu, M.-Y., Zhu, J.-Y., Liu, G., Tao, A., Kautz, J., and Catanzaro, B. (2018c). Video-to-video synthesis. In *Advances in Neural Information Processing Systems*, pages 1144–1156.

[Wang et al., 2019b] Wang, T.-C., Liu, M.-Y., Tao, A., Liu, G., Catanzaro, B., and Kautz, J. (2019b). Few-shot video-to-video synthesis. In *Advances in Neural Information Processing Systems*, pages 5013–5024.

[Wang et al., 2017] Wang, W., Huang, Q., You, S., Yang, C., and Neumann, U. (2017). Shape inpainting using 3d generative adversarial network and recurrent convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2298–2306.

[Wang and Gupta, 2016] Wang, X. and Gupta, A. (2016). Generative image modeling using style and structure adversarial networks. In *Proceedings of the European Conference on Computer Vision*, pages 318–335. Springer.

[Wang and Mori, 2008] Wang, Y. and Mori, G. (2008). Multiple tree models for occlusion and spatial constraints in human pose estimation. In *Proceedings of the European Conference on Computer Vision*, pages 710–724. Springer.

[Wang et al., 2011] Wang, Y., Tran, D., and Liao, Z. (2011). Learning hierarchical poselets for human parsing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1705–1712. IEEE.

[Wang and Ye, 2015] Wang, Z. and Ye, J. (2015). Querying discriminative and representative samples for batch mode active learning. In *Transactions on Knowledge Discovery from Data*, volume 9, pages 158–166. Association for Computing Machinery.

[Ward Jr, 1963] Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58 :236–244.

[Wei et al., 2016] Wei, S.-E., Ramakrishna, V., Kanade, T., and Sheikh, Y. (2016). Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732.

[Wei et al., 2012] Wei, X., Zhang, P., and Chai, J. (2012). Accurate realtime full-body motion capture using a single depth camera. *ACM Transactions on Graphics*, 31(6) :1–12.

[Werbos, 1974] Werbos, P. J. (1974). *New tools for prediction and analysis in the behavioral sciences*. PhD thesis, Harvard University.

[Wikipédia, 2019] Wikipédia (2019). Membre supérieur (anatomie humaine). https://fr.wikipedia.org/wiki/Membre_sup%C3%A9rieur_(anatomie_humaine).

[Wixted et al., 2018] Wixted, F., Shevlin, M., and O'Sullivan, L. W. (2018). Distress and worry as mediators in the relationship between psychosocial risks and upper body musculoskeletal complaints in highly automated manufacturing. *Ergonomics*, 61(8) :1079–1093.

[Wu et al., 2019] Wu, H., Zheng, S., Zhang, J., and Huang, K. (2019). Gp-gan : Towards realistic high-resolution image blending. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2487–2495. Association for Computing Machinery.

[Wu et al., 2016] Wu, J., Zhang, C., Xue, T., Freeman, B., and Tenenbaum, J. (2016). Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90.

[Xiao et al., 2018] Xiao, B., Wu, H., and Wei, Y. (2018). Simple baselines for human pose estimation and tracking. In *Proceedings of the European Conference on Computer Vision*, pages 466–481.

[Xie et al., 2020] Xie, Y., Jiaojiao, T., and Zhu, X. (2020). Linking points with labels in 3d : A review of point cloud semantic segmentation. *IEEE Geoscience and Remote Sensing Magazine*.

[Xiu et al., 2018] Xiu, Y., Li, J., Wang, H., Fang, Y., and Lu, C. (2018). Pose flow : Efficient online pose tracking. In *British Machine Vision Conference*. British Machine Vision Association.

[Yang et al., 2016] Yang, W., Ouyang, W., Li, H., and Wang, X. (2016). End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3073–3082.

[Yang et al., 2017] Yang, W., Li, S., Ouyang, W., Li, H., and Wang, X. (2017). Learning feature pyramids for human pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1281–1290.

[Yang and Ramanan, 2011] Yang, Y. and Ramanan, D. (2011). Articulated pose estimation with flexible mixtures-of-parts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1385–1392. IEEE.

[Yang and Ramanan, 2013] Yang, Y. and Ramanan, D. (2013). Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(12) :2878–2890.

[Yasin et al., 2016] Yasin, H., Iqbal, U., Kruger, B., Weber, A., and Gall, J. (2016). A dual-source approach for 3d pose estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4948–4956.

[Ye et al., 2011] Ye, M., Wang, X., Yang, R., Ren, L., and Pollefeys, M. (2011). Accurate 3d pose estimation from a single depth image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 731–738. IEEE.

[Ye and Yang, 2014] Ye, M. and Yang, R. (2014). Real-time simultaneous pose and shape estimation for articulated objects using a single depth camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2345–2352.

[Yezzi et al., 1997] Yezzi, A., Kichenassamy, S., Kumar, A., Olver, P., and Tannenbaum, A. (1997). A geometric snake model for segmentation of medical imagery. *IEEE Transactions on Medical Imaging*, 16(2) :199–209.

[Yi et al., 2017] Yi, Z., Zhang, H., Tan, P., and Gong, M. (2017). Dualgan : Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2849–2857. IEEE.

[Yu et al., 2018] Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., and Sang, N. (2018). Bisenet : Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European Conference on Computer Vision*, pages 334–349. Springer.

[Yu et al., 2020] Yu, C., Gao, C., Wang, J., Yu, G., Shen, C., and Sang, N. (2020). Bisenet v2 : Bilateral network with guided aggregation for real-time semantic segmentation. *arXiv preprint arXiv :2004.02147*.

[Yub Jung et al., 2015] Yub Jung, H., Lee, S., Seok Heo, Y., and Dong Yun, I. (2015). Random tree walk toward instantaneous 3d human pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2467–2474. IEEE.

[Zhang et al., 2017a] Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. N. (2017a). Stackgan : Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915.

[Zhang et al., 2017b] Zhang, L., Ji, Y., Lin, X., and Liu, C. (2017b). Style transfer for anime sketches with enhanced residual u-net and auxiliary classifier gan. In *Proceedings of the 4th IAPR Asian Conference on Pattern Recognition*, pages 506–511. IEEE.

[Zhang et al., 2019] Zhang, X., Wong, Y., Kankanhalli, M. S., and Geng, W. (2019). Unsupervised domain adaptation for 3d human pose estimation. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 926–934. Association for Computing Machinery.

[Zhang et al., 2001] Zhang, Y., Brady, M., and Smith, S. (2001). Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE Transactions on Medical Imaging*, 20(1) :45–57.

[Zheng et al., 2009] Zheng, L., Zhang, J., and Wang, Q. (2009). Mean-shift-based color segmentation of images containing green vegetation. *Computers and Electronics in Agriculture*, 65(1) :93–98.

[Zhou et al., 2004] Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, pages 321–328.

[Zhou et al., 2016a] Zhou, X., Sun, X., Zhang, W., Liang, S., and Wei, Y. (2016a). Deep kinematic pose regression. In *Proceedings of the European Conference on Computer Vision*, pages 186–201. Springer.

[Zhou et al., 2016b] Zhou, X., Zhu, M., Leonardos, S., Derpanis, K. G., and Daniilidis, K. (2016b). Sparseness meets deepness : 3d human pose estimation from monocular video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4966–4975. IEEE.

[Zhou et al., 2017] Zhou, X., Huang, Q., Sun, X., Xue, X., and Wei, Y. (2017). Towards 3d human pose estimation in the wild : a weakly-supervised approach. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 398–407.

[Zhou et al., 2018] Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N., and Liang, J. (2018). Unet++ : A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*, pages 3–11. Springer.

[Zhou et al., 2009] Zhou, Z.-h., Sun, Y.-y., and Li, Y.-f. (2009). Multi-instance learning by treating instances as non-i.i.d. samples. In *Proceedings of the 26th International Conference on Machine Learning*, pages 1249–1256. Association for Computing Machinery.

[Zhou, 2017] Zhou, Z.-H. (2017). A brief introduction to weakly supervised learning. *National Science Review*, 5 :44–53.

[Zhu et al., 2016] Zhu, J.-Y., Krähenbühl, P., Shechtman, E., and Efros, A. A. (2016). Generative visual manipulation on the natural image manifold. In *Proceedings of the European Conference on Computer Vision*, pages 597–613. Springer.

[Zhu et al., 2017] Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232.

[Zhu, 2006] Zhu, X. J. (2006). Semi-supervised learning literature survey. *University of Wisconsin-Madison Department of Computer Sciences*, 2.

[Zhu and Fujimura, 2007] Zhu, Y. and Fujimura, K. (2007). Constrained optimization for human pose estimation from depth sequences. In *Asian Conference on Computer Vision*, pages 408–418. Springer.

[Zhu et al., 2008] Zhu, Y., Dariush, B., and Fujimura, K. (2008). Controlled human pose estimation from depth image streams. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8. IEEE.

[Zhu and Fujimura, 2010] Zhu, Y. and Fujimura, K. (2010). A bayesian framework for human body pose tracking from depth image sequences. *Sensors*, 10(5) :5280–5293.

[Zimmer et al., 2002] Zimmer, C., Labruyere, E., Meas-Yedid, V., Guillén, N., and Olivo-Marin, J.-C. (2002). Segmentation and tracking of migrating cells in videomicroscopy with parametric active contours : A tool for cell-based drug testing. *IEEE Transactions on Medical Imaging*, 21(10) :1212–1221.

[Zuffi and Black, 2015] Zuffi, S. and Black, M. J. (2015). The stitched puppet : A graphical model of 3d human shape and pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3537–3546.

Popularized abstract

Aiming to study stress and ergonomics in the workplace, the Ebhys company wants to develop a tool to analyse the activity of human operators in a waste sorting centre, by measuring ergonomic indicators. These indicators are computed using depth frames, which are images containing the 3D information of the scene that they capture.

An ergonomic study allows us to define several indicators of stress and ergonomics, which are based on the operator's pose. Thus, the activity analysis system is divided into three modules : a first module that extracts the operator from the depth image, a second module that computes the operator's posture and a third module that gives the ergonomic indicators based on the operator's posture.

To build the operator extraction and posture computation modules, we propose innovative algorithms to cope with the difficulty induced by the deployment of the system in an industrial environment. Specifically, these innovatives algorithms use an intelligent selection process of relevant images and synthetically generated depth images.

Résumé vulgarisé

Dans un contexte d'étude de la pénibilité et de l'ergonomie au travail, la société Ebhys cherche à développer un outil d'analyse de l'activité des opérateurs humains dans un centre de tri de déchets, en mesurant des indicateurs ergonomiques. Ces indicateurs sont mesurés à partir d'images de profondeur, qui sont des images contenant une information 3D de la scène qu'elles représentent.

Une étude ergonomique nous permet de définir plusieurs indicateurs de pénibilité, basés sur la posture de l'opérateur. Ainsi, le système d'analyse de la pénibilité se découpe en trois modules : un premier module qui extrait l'opérateur de l'image de profondeur, un second module qui calcule sa posture et un troisième module qui donne les indicateurs ergonomiques à partir de la posture de l'opérateur.

Pour construire les modules d'extraction de l'opérateur et de calcul de la posture, nous proposons des algorithmes innovants pour faire face à la difficulté provoquée par le déploiement du système dans un environnement industriel. Ces algorithmes ont recours à la sélection intelligente d'images pertinentes et à l'utilisation d'images de profondeur de synthèse générées numériquement.

Abstract

In a context of study of stress and ergonomics at work for the prevention of musculoskeletal disorders, the company Ebhys wants to develop a tool for analyzing the activity of human operators in a waste sorting center, by measuring ergonomic indicators. To cope with the uncontrolled environment of the sorting center, these indicators are measured from depth images.

An ergonomic study allows us to define the indicators to be measured. These indicators are zones of movement of the operator's hands and zones of angulations of certain joints of the upper body. They are therefore indicators that can be obtained from an analysis of the operator's 3D pose. The software for calculating the indicators will thus be composed of three steps : a first part segments the operator from the rest of the scene to ease the 3D pose estimation, a second part estimates the operator's 3D pose, and the third part uses the operator's 3D pose to compute the ergonomic indicators.

First of all, we propose an algorithm that extracts the operator from the rest of the depth image. To do this, we use a first automatic segmentation based on static background removal and selection of a moving element given its position and size. This first segmentation allows us to train a neural network that improves the results. This neural network is trained using the segmentations obtained from the first automatic segmentation, from which the best quality samples are automatically selected during training.

Next, we build a neural network model to estimate the operator's 3D pose. We propose a study that allows us to find a light and optimal model for 3D pose estimation on synthetic depth images, which we generate numerically. However, if this network gives outstanding performances on synthetic depth images, it is not directly applicable to real depth images that we acquired in an industrial context.

To overcome this issue, we finally build a module that allows us to transform the synthetic depth images into more realistic depth images. This image-to-image translation model modifies the style of the depth image without changing its content, keeping the 3D pose of the operator from the synthetic source image unchanged on the translated realistic depth frames. These more realistic depth images are then used to re-train the 3D pose estimation neural network, to finally obtain a convincing 3D pose estimation on the depth images acquired in real conditions, to compute de ergonomic indicators.

Résumé

Dans un contexte d'étude de la pénibilité et de l'ergonomie au travail pour la prévention des troubles musculo-squelettiques, la société Ebhys cherche à développer un outil d'analyse de l'activité des opérateurs humains dans un centre de tri, par l'évaluation d'indicateurs ergonomiques. Pour faire face à l'environnement non contrôlé du centre de tri et pour faciliter l'acceptabilité du dispositif, ces indicateurs sont mesurés à partir d'images de profondeur.

Une étude ergonomique nous permet de définir les indicateurs à mesurer. Ces indicateurs sont les zones d'évolution des mains de l'opérateur et d'angulations de certaines articulations du haut du corps. Ce sont donc des indicateurs obtenables à partir d'une analyse de la posture 3D de l'opérateur. Le dispositif de calcul des indicateurs sera donc composé de trois parties : une première partie sépare l'opérateur du reste de la scène pour faciliter l'estimation de posture 3D, une seconde partie calcule la posture 3D de l'opérateur, et la troisième utilise la posture 3D de l'opérateur pour calculer les indicateurs ergonomiques.

Tout d'abord, nous proposons un algorithme qui permet d'extraire l'opérateur du reste de l'image de profondeur. Pour ce faire, nous utilisons une première segmentation automatique basée sur la suppression du fond statique et la sélection d'un objet dynamique à l'aide de sa position et de sa taille. Cette première segmentation sert à entraîner un algorithme d'apprentissage qui améliore les résultats obtenus. Cet algorithme d'apprentissage est entraîné à l'aide des segmentations calculées précédemment, dont on sélectionne automatiquement les échantillons de meilleure qualité au cours de l'entraînement.

Ensuite, nous construisons un modèle de réseau de neurones pour l'estimation de la posture 3D de l'opérateur. Nous proposons une étude qui permet de trouver un modèle léger et optimal pour l'estimation de posture 3D sur des images de profondeur de synthèse, que nous générons numériquement.

Finalement, comme ce modèle n'est pas directement applicable sur les images de profondeur acquises dans les centres de tri, nous construisons un module qui permet de transformer les images de profondeur de synthèse en images de profondeur plus réalistes. Ces images de profondeur plus réalistes sont utilisées pour réentrainer l'algorithme d'estimation de posture 3D, pour finalement obtenir une estimation de posture 3D convaincante sur les images de profondeur acquises en conditions réelles, permettant ainsi de calculer les indicateurs ergonomiques.