Embodied robot systems : control and learning for skill-transfer from humans

Guillaume Gourmelen

THÈSE POUR OBTENIR LE GRADE DE DOCTEUR DE L'UNIVERSITE DE MONTPELLIER

En Systèmes Automatiques et Microélectroniques

École doctorale : Information, Structures, Systèmes

Unité de recherche UMR5506

Embodied robot systems : control and learning for skill-transfer from humans

Présentée par Gourmelen Guillaume

Le 29 Novembre 2022

Sous la direction de Gowrishankar Ganesh et Cherubini Andrea

Devant le jury composé de

Azevedo Christine, Directrice de Recherche, INRIA	Examinatrice
Burdet Etienne, Professeur d'Université, Imperial College of Science, Technology and Medicine	Rapporteur
Gaussier Philippe, Professeur d'Université, CY Paris Cergy Université	Rapporteur
Gotlieb Arnaud, Directeur de Recherche, Simula Research Laboratory	Président du jury
Gowrishankar Ganesh, Directeur de Recherche, CNRS-LIRMM	Directeur de thèse
Cherubini Andrea, Professeur d'Université, Université de Montpellier-LIRMM	Encadrant de thèse

UNIVERSITÉ DE MONTPELLIER

ACKNOWLEDGEMENTS

I would like to thank :

The jury members for reading my manuscript, and assisting to my Defence.

My supervisor Ganesh Gowrishankar for giving me the chance to accomplish this thesis for 3 years and 4 months.

Adrien Verhulst for the Co-limb shooting, setup and the precious help given during my first year.

Yukiko Iwasaki for her help throughout my PhD and for all the good time we had as office-mates.

Sasaki Tomoya, Ando Kozo, Iizuka Shuhei for their help before and during Siggraph Asia, without the complete team it would have been impossible to bring Metalimb in Brisbane, build it in the booth and manage all the hardware problems the happened there.

Matsumura-san from Karakuri Product for his help during the weeks I spent in his office in Kanagawa.

Inami-sensei for accepting me in his lab almost 3 months in 2019 and the Tokyo University Inami living lab secretary and students who welcomed me every-time.

Benjamin Navarro for his precious help on the robotic part of this thesis when he was a post doc in the team and everything he taught me in order to understand programming languages, ROS and the demystification of all the robotics equations we needed to use in order to have the robots work.

The regular players of the lab board game club, our hiking team and the people from our "potager" garden, Celia S and Silvia C for always being supportive.

Quentin M and Thomas G, old time friends, for always being here on the campus when in need to talk.

Maxime M for your time he accorded to me, despite your own manuscript redaction, to discuss Deep Learning and debug Tensor Flow based networks.

Yuquan Wang for all the discussion we had about academic research we had during these long Saturday and weeknight of work in the empty lab and your help as a team post-doc.

Amaury Dechaux for your knowledge in data-science and statistics.

Virginie Feche who helped me wholeheartedly in organizing all the event we featured for the PhD Students like the IA teaching days, the PhD poster sessions and seminaries, the PhD Discord Channel, the article writing sessions, and all the buying we had to do.

Gwladys and Nadine from the LIRMM who helped me in managing all the trips I made to Tokyo so I could spend time at Tokyo University, then supported me in keeping lab access and solving administrative problems during the 4th year when I was not funded.

Vincent Berry and Eric Dubreuil for trusting me to teach around an hundred hour of classes and experiments to the Polytech students from "Informatique et Gestion" and "Systèmes Embarqués" departments

My parents and my brother to have been here for me throughout these 9 years of University.

Personal feedback about the Co-limb project Personally, this project and in particular preparing Siggraph Asia, taught me to work with a team as I almost had no other teamwork opportunity during this thesis. Moreover this project being held in collaboration with Tokyo University, I was mainly working with Japanese people, in Japanese, as all students included in the project did not speak English. The two last weeks before the conference I also had the opportunity to work at Karakuri, the company that developed Meta-Limbs, in order to speed up all the modifications and last minute tests needed for Siggraph. This gave me an interesting insight of the way of working abroad in Japan in both academia and in small companies.

RÉSUMÉ DE LA THÈSE

La plupart des robots sont meilleurs que les humains en termes de capteurs et d'actionneurs, ce qui leur permet d'avoir une meilleure perception, des mouvements plus rapides et une force supérieure à celle des humains. Cependant, il leur manque la capacité humaine à prendre des décisions intelligentes de haut niveau dans des environnements non structurés. Par conséquent, de nombreux systèmes complexes actuels, tels que les robots de sauvetage, les UAV/UUV et les robots d'assistance chirurgicale, utilisent des systèmes "humains dans la boucle", où le contrôle est partagé entre l'humain et la machine. Cette thèse explore les "robots incarnés" comme un phénomène prometteur pour améliorer le contrôle humain dans la boucle.

Par robots incarnés, nous nous référons aux robots attachés à l'opérateur humain, et qui sont contrôlés par l'opérateur humain en utilisant des retours d'informations (principalement visuel et haptique) qui lui font percevoir le robot comme une partie de lui-même.

Nous avons tout d'abord travaillé avec un dispositif de bras de robot portable disponible pour concevoir un contrôleur d'admitance qui permettrait aux opérateurs humains de le contrôler et de l'utiliser intuitivement.

Deuxièmement, nous avons développé un système de téléopération incarné (Embodied) pour un manipulateur sériel 7DoF en intégrant le système robotique avec un dispositif de retour haptique et un casque de réalité virtuelle couplé à une caméra à 360 degrés.

Troisièmement, pour ce système, nous avons mis au point une procédure permettant d'estimer l'impédance de l'opérateur humain sans avoir recours à l'électromyographie, et de l'utiliser pour contrôler l'impédance du robot, ce qui permet aux utilisateurs d'imposer à la fois des mouvements et une impédance au robot.

Dernièrement, nous avons étudié l'apprentissage par démonstration en utilisant des informations visuelles et de mouvement humain à travers le système incarné afin pouvoir entrainer un agent à reproduire des mouvements humains.

Nous postulons que les systèmes incarnés peuvent améliorer l'apprentissage par la démonstration en permettant un enseignement kinesthésique sans perte du point de vue de l'opérateur et de la sensation de force. En effet, dans de tels systèmes, il est possible d'enregistrer tous les mouvements, toutes ce que a été vu et toutes les sensations d'un être humain, ce qui en fait un outil idéal pour la mise en œuvre de techniques d'apprentissage. Pour démontrer cette idée, nous avons développé une tâche demandant à des participants humains de déplacer un curseur dans un labyrinthe en pré-

sence de différents champs de force. Nous avons ensuite entraîné un agent de réseau de neuronne sur ces comportements. En utilisant des réseaux tels que ResNet(Residual Networks) et des Gated Recurrent Units, l'agentpeut apprendre des mouvements semblables à ceux des humains, de les généraliser et de prédire le comportement humain dans des trajectoires sur lesquelles il n'avait jamais été formé.

Nous concluons ce travail avec une discussion portant sur les résultats obtenus et les perspectives pour la recherche dans les domaines du controle de robot incarnés et de l'apprentissage par démonstration.

ABSTRACT

Most robots are better than humans in terms of sensors and actuators, which allow them to have higher and better perception, and faster movements and superior strength, than humans. However, what they lack is the human ability to make intelligent high level decisions in unstructured environments. Therefore, many current complex systems, like rescue robots, UAV/UUVs, and surgery assistive robots, utilize 'human in the loop' systems, where the control is shared by the human and machine. This PhD explores 'embodied robots' as a promising phenomenon to improve human-in-the-loop control.

By embodied robots, we refer to robots attached to the human operator, and which are controlled by the human operator using first person feedback (predominantly visual and haptic) that make him/her perceive the robot to be part of his/her self.

First, we worked with an available Wearable Robot Arm (WRA) device to design an admittance controller that would let human operators control and use it intuitively.

Second, we developed an embodied teleoperation system for a 7DoF serial manipulator by integrating the robot system with a haptic feedback device, and virtual reality head mounted display coupled to a 360 degree camera.

Third, for this system, we developed a procedure to estimate the impedance of the human operator without requirering electromyography, and use it for impedance control of the robot, enabling users to impose both movements as well as impedance on the robot slave.

Finally, we studied Learning from Demonstration (LfD) using visual and human motion data through our embodied system. We postulate that embodied systems can improve LfD by enabling kinesthetic teaching without losing of the operator's point of view and force sensation. Indeed in such systems it is possible to record every motion, vision and feeling the human would have, making it ideal for the implementation of machine learning techniques. To demonstrate this idea, we developed a task requiring human participants to move through a maze in the presence of various force fields. We then trained a deep learning network agent on these behaviors. By using a state of the art pre-trained Residual Networks and Gated Recurrent Units, the machine learning agent was able to learn 'human-like' movements and generalize, and predict human behavior in trajectories that it was never trained on.

We conclude this work with a discussion of the achieved results and future perspectives of research in the field of embodied control and Learning from Demonstration.

Keywords

Human in the loop control, tele-impedance, embodiment, embodied tele-operation, deep learning, learning from demonstration, learning from observation

Contents

Nomenclature

- ANN : Artificial Neural Networks
- CNN : Convolutional Neural Network
- DoF : Degree of Freedom
- EMG : Electromyography
- GMM : Gaussian Mixture Models
- GRU : Gated Recurrent Unit
- HMD : Head Mounted Display
- HRI : Human Robot Interaction
- IRL : Inverse Reinforcement Learning
- LfD : Learning from Demonstration
- LSTM : Long Short Term Memory
- MDN : Mixture Density Network
- MLP : MultiLayer Perceptron
- NaN : Not a Number
- ResNet : Residual Network (pretrained)
- RL : Reinforcement Learning
- RHI : Rubber Hand Illusion
- RNN : Recurrent Neural Networks
- TCP : Terminal Control Point
- UAV : Unmanned Aerial Vehicle
- UUV : Unmanned Underwater Vehicle
- VR : Virtual Reality
- WRA : Wearable Robot Arm

INTRODUCTION

A century ago authors like Capek, Asimov, described the future of humanity with robots alongside humans, would that be in utopian or dystopian works. The word 'Robot' was introduced by Capek in RUR in which he described a society where all the work is done by robots for the human's sake. This idea comes from the greek philosophy that thought that citizens would become free when they would stop working, leaving all the work to their slaves. These human-like, anthropomorphic machines have been found through the past few centuries : the automatons of Leonardo da Vinci in the 15th century, the japanese puppet theatre called Bunraku in the 17th century are few examples. But all these inventions have one common point: they could be controlled by a human, from a remote place. And this point has not changed so much through time.

"Autonomous" robots that require not much guidance from human are mainly found in strictly controlled or modified environments, mainly in industries or warehouses. The only "autonomous" robots present in human environments at a large scale are vacuum cleaning robots and alike, thanks to the recent advances of localization and path planning like SLAM algorithms. Crucially, there are almost no autonomous robots that can interact physically with humans, and in fact they are smoothly separated from humans.

Most current robots capable of physically interacting with their environment, and with humans are invariably teleoperated, with some human in the loop control. For example, the submarine manipulator robots like Ocean One [Khatib et al., 2016], surgical robots (Da Vinci Surgical, Rosa Knee), co-bots, rescue robots [Liu and Nejat, 2013] and even space robot [Papadopoulos et al., 2021]

Hence, human-in-the-loop control is still being used for complex task achievement like rescue robots, collaborative robots fr industry, most UAV/UUV operation, or robot-assisted surgery. Understanding that robotic systems will not work without human assistance for at least a few years more from now on, in this thesis we decided to explore the embodiment phenomenon to improve human-in-the-loop control.

The thesis, and my work was part of an international Japanese project in which the group of Dr. Ganesh Gowrishankar from CNRS is a partner. The Inami Jizai Erato Project was a 5 years project funded by the Japan Science and Technology (JST) institute. The project takes an inter-disciplinary approach to understand, and address the challenges of technology to "extend" the human body. The project involved teams working in Design, Virtual Reality, robotics, neuroscience and psychology and spanned several major Japanese universities including The University of Tokyo, Waseda University,

Keio University, The University of Electro-Communications, Toyohashi University of Technology, as well as the Centre National de la Recherche Scientifique (CNRS), and S Care Design Lab Japan.

In this thesis, we refer to a robotic human in loop system as "embodied" when the robot is controlled by a human user with a first person visual point of view. The robot is commanded by tracking of any limb of the human user, with the robot's location matching, in first person perspective, the location of the human limb that the robot tracks. The user is provided with relevant haptic information during in the task, and where he/she would have felt the feedback in case he had used his own limb to perform the task. Overall, the human user would feel as if an embodied robot was part of him/her and replaced the limb/s used to control the robot .

In this thesis we will focus on the issues of embodiment on tele-operation and the benefits and constraints of embodiment in regard to the control of and learning by robots. We do not aim to improve other common problems in teleoperation, especially delays, which we will consider as solved, or as a constraint in our investigations.

In this thesis, we will start with chapters on the Background and State-of-the-Art where we will define backgrounds of the main concepts underlying this thesis. Specifically, we will discuss some background information on serial manipulator control, haptic feedback device, teleoperation, tele-impedance, embodiment phenomenon, embodied robot, deep learning and Learning from Demonstration

Then in the second chapter, I will present my work to develop an intuitive controller and user interface for an embodied wearable system with two robot arms. The work was performed in collaboration with the University of Tokyo and Published as an E-tech paper at Siggraph Asia 2019, which is one of the two leading (and strictly reviewed) conference in the field of Graphics and Interactive techniques.

In the third chapter, I will then present the embodied teleoperation robot setup and control, that I developed by integrating several hardware elements and software environment. This setup was the base for the tele-impedance experiments and the Learning from Demonstration experiments, which form a major part of this thesis.

The fourth chapter presents our implementation of a new human arm Impedance estimation algorithm, and the implementation of tele-impedance in the embodied teleoperation setup. This work was presented at the IEEE Intelligent Robots and Systems (IROS) 2021.

Chapters 5 and 6 will then present my work on the implementation of an embodied skill transfer system with our system. I discuss how an embodied tele-operation setup is ideal in many respect for the implementation of the machine leaning based algorithms to learn from a human and improve a robot's skill. Chapter 5 will introduce background on Machine and Deep Learning aspects to understand the choices made in Chapter 6.

The Chapter 6 will then describe our Deep Learning-based architecture based on visual recognition and our experiment environment. Then it will explain the results obtained with the trained agent and will present a comparison of our results with hu-

man participants.

Finally, Chapter 7 discussion the thesis contribution and give a perspective for future developments.

Note about the QR Codes

This thesis contains some QR codes pointing to our Git repository. This is a complement to the figure that represents those motions. All the QR codes are either flash-able by your smartphone to access the link or clickable to directly open the link in your internet browser.

CHAPTER 1

BACKGROUND AND STATE OF THE ART

Contents

This thesis concerns embodied robot systems and skill transfer through these systems. We define embodiment, and describe the link between robotics and embodiment. Embodied robots also need to be controlled, and teleoperated, with haptic feedback devices to achieve embodiment. Furthermore, we also introduce Learning from Demonstration in embodied robots, including Machine Learning and Deep Learning in robotics or in robotic-related field.

1.1 Teleoperation

The word Teleoperation appeared in dictionaries only around 1975 but it was used since before World War 2, when scientists started understanding that radioactive products are harmful to the human body. The need to manipulate these products for research purposes from behind lead walls led to building some primitive mechanical tele-operation systems. The first models were just mechanical-ball joints and translating links moving trough a lead wall, letting the teleoperator manipulate with four Degree of Freedom with the help of a long clamp.

The first Master Slave teleoperation system was proposed by Raymond Goertz in 1947 in Chicago (Fig. 1.1). It was a mechanical teleoperation device where the teleoperation was bilateral. Goertz would later adapt this concept to electrically operated actuators in order to get more remote teleoperation but lose bilaterality.

As any teleoperation system has two sides, in this thesis from now on, the human side will be called the teleoperator or the master and the robot side called the slave or the manipulator system.

The word teleoperation also gave birth to the words tele-existence, tele-robotics, tele-presence, tele-manipulation, tele-impedance over the time which are all teleoperation tasks but each has a specific meaning. Teleoperation has been used in many devices such as for micromanipulation [Bolopion and Regnier, 2013], trough the Internet communication network [Kebria et al., 2018], for arm robot mounted on UAV [Leica et al., 2019], for Manipulation in space [Weber et al., 2019, Imaida et al., 2004],etc..

In this thesis we do not treat teleoperation aspects such as stability, time delay concerns, passivity. We used teleoperation for teleimpedance purposes as described in Section 1.5 in a bilateral teleoperation system. In order to achieve bilaterality, we used the haptic feedback device presented in next Section.

FIGURE 1.1 – *Raymond Goertz and his first teleoperation system, Source : Wikipedia Commons*

1.2 Haptic feedback

Haptic feedback can be defined as the forces sensed or vibrations coming from any device in reaction to the user action or environment changes. The word haptic comes from a Greek word haphtésaï meaning feeling by touching. This sense combines 2 different type of information : the kinestetic or proprioception senses and the tactile information should they be forces, shapes, textures or thermal exchanges. Haptic feedback technologies are used in multiples fields, including video games, Virtual Reality, Ro-

botics, Robotics Surgeries, Virtual Training, race simulations, planes simulations...

Haptic feedback usages can be separated in 2 categories :
-making the user feel a non physical, virtual, feedback
-reproducing physical signals to a remote location, for teleoperation purposes or amplification purposes.

There are two mains type of haptic feedback devices :
-the ones that only give feedback
-the ones that beside sending feedback can also read some of the user's parameters, like a relative pose, velocity etc...

For VR both types can be used but for the robotic and teleoperation uses only the later is generally useful as it can generate the desired command to the robot. Haotic devices are also present in multiple devices, from video games controller or smartphones that will simply vibrate depending on the user action, to complex 6DoF haptic feedback devices. Indeed in case of robotics the feedback also depends on the position of the physical interaction. So as a user you might receive a constant force feedback, but feel a different force when the robot achieves some physical interaction.
In our experiment we use haptic feedback during teleoperation to have the user feel forces created by physical interaction between the robot and its environment while letting the user control the robot intuitively. To control the robot we used impedance control as we will detail in the next section

1.3 Impedance control in serial manipulators

Serial manipulators can be controlled in multiple ways. The most popular in the industry being position control or speed control. They consist in computing the position or angular speed of each joint to lead to the given position or speed of the Terminal Control Point (TCP). These methods are generally stable, precise and robust to external disturbances. And this is exactly where the limit is reached. Indeed, if a contact were to be made with the environment, there would be no compliance. The robot controller would likely reach the desired command and an impact would likely occur. This is especially problematic in tasks where the environment is not accurately well known and for Human-Robot Interaction (HRI) since any impact would mean potential safety issues for the human.

In order to achieve safely this kind of task, impedance control is generally used and has made its proof since almost four decades [Hogan, 1984, Almeida et al., 1999, Jung et al., 2004]. A torque-controlled enabled robot is needed to have a real, non simulated, impedance control. It implies that each joints of the serial arm is equipped with a torque sensor embedded and that each DC motor can be torque controlled. It is also beneficial to have a force sensor mounted on the robot TCP measuring external forces or an estimation of external forces, directly by using the embedded torque sensors.

In the case of impedance control, the command is not anymore a desired pose but a desired impedance. The robot TCP is generally modeled as a mechanical mass-spring-

"Haptic Gloves" from Manus VR	"Omega" from Force dimension
	
"TorsionCrowds" from [Horie et al., 2020]	"Haption Virtuose 6D" from Haption
	
"Hardlight VR" Haptic Suit	"Phantom Sensable" from Horie and al
	

TABLE 1.1 – *The 2 bottom-most left column devices are haptic feedback "only" devices. The right column devices and left first device are haptic feedback device that can record the user's position, pose and/or velocity*

damper system in the Task-Space with the following equation (1.1):

$$\boldsymbol{F^R} = \boldsymbol{K}\Delta\boldsymbol{x^R} + \boldsymbol{D}\Delta\dot{\boldsymbol{x}}^R + \boldsymbol{M}\Delta\ddot{\boldsymbol{x}}^R, \tag{1.1}$$

where $\boldsymbol{F^R} \in \mathbb{R}^6$ are the external forces applied to the robot TCP , $\boldsymbol{K}$, $\boldsymbol{D}$ and $\boldsymbol{M}$ are the positive-definite 6×6 matrices of stiffness, damping and mass, and $\Delta\boldsymbol{x^R} = \boldsymbol{x^R}_r - \boldsymbol{x^R} \in \mathbb{R}^6$ is the difference between the reference and measured positions of the robot TCP

With impedance control, impedance values like stiffness and damping matrix would be estimated for the controller to work. In general, the stiffness matrix $\boldsymbol{K}$ is chosen depending on the need of the task, with an empirical process of trial an error to tune the value and damping set to a value that would lead to an overdamped system so the system stays stable. We explain how human arm impedance can be estimated the next section.

1.4 Human impedance estimation

During impedance control the impedance values to be set to the robot should be decided, and in some cases there is a need to change the robot impedance online. Since our work is focusing on human-in-the-loop system, we decided to apply impedance parameters taken from the human on the robot. In order to achieve this type of control called tele-impedance (Section 1.5) we first need to estimate the teleoperator's arm impedance.

Human impedance estimation in regard to robotics has been popularly done through EMG (Electromyography) sensors directly placed on the skin of the human user and measure the electrical activity of the muscles. [Ganesh et al., 2010, Peternel et al., 2017, Luo et al., 2019, Ajoudani et al., 2012]. In the field of teleoperation when robots are impedance-controlled during teleoperation, the operator arm impedance can be used to control intuitively the robot's impedance. In these case, EMG sensors can be placed on different muscles of the arm. (Fig. 1.2). To estimate the human arm impedance even if no external forces are applied on the human arm, hence co-contractions without physical external contact are measurable.

The drawback of this method is that there are multiples sensors, hence data should be treated and merged, and calibrated as it requires user-specific calibration [Ajoudani, 2016, Doornebosch et al., 2021] and it is also noisy. EMG also include muscle reflexes, which are characterized by their own temporal and state dynamics [Doornebosch et al., 2021] making impedance estimation non trivial. Less known methods also exist, like measuring the force in the user's hand grasp force because naturally the human tends to close his hand when co-contracting muscle [Walker et al., 2010] but it only estimates in 1-Dimension.

Another method consists in adding a vibrating device on the human wrist and by knowing the vibration and by measuring the reaction in position, velocity and acceleration of human arm, it is possible to estimate impedance.[Gomi and Osu, 1996, 1998, Burdet et al., 2001, Hill and Niemeyer, 2009]

We decided not to use EMG due to the aforementioned drawbacks and developed an online perturbation based impedance estimation methodology. That is, our method uses the perturbations inherently present in the task to estimate the human's arm impedance. This is described in Chapter 4.

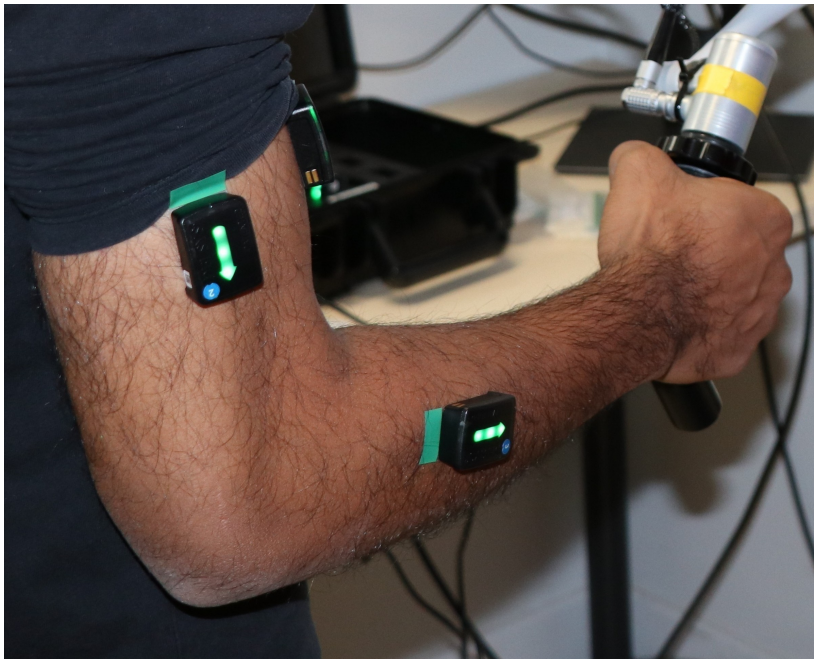


FIGURE 1.2 – *EMG sensors fixed to a participant skin during one of our experiment*

1.5 Tele-impedance

We call Tele-impedance a teleoperation where the slave robot is impedance controlled and the master in the system has not only his/her movements sent trough the teleoperation but also his/her arm impedance parameters estimated. In the case the user's impedance is estimated in real time, this impedance estimation is then applied to the slave robot, so the robot reacts in the same mechanical way as the human operator does, in position, velocity, acceleration and stiffness and damping. Tele-impedance is being used in field like HRI or industrial tasks [Ajoudani, 2016, Doornebosch et al., 2021]

In our work we use tele-impedance to improve our embodied setup. Indeed the user can control the robot impedance in real time, which enables more possibilities about task achievement but also improves Human Robot Interactions through an embodied robot. The method used to estimate human arm impedance and apply it on the robot impedance control, is described in Chapter 4.

1.6 Embodiment

Embodiment or more specifically the sense of embodiment in a human being has been defined as "the ensemble of sensations that arise in conjunction with being inside, having, and controlling a body" ([Kilteni et al., 2012]), where "body" refers to a physical or virtual body, not only a biological one. While this definition sounds complex, it is quite simple to have a participant feel the sense of embodiment. The best known experiment is the Rubber Hand Illusion (RHI) (Fig. 1.3) achieved by [Botvinick and Cohen, 1998]. While a participant sits at a table a left rubber hand is placed at the right to his/her real left hand, with an opaque screen preventing the participant from seeing his/her own arm and hand. When done, the experimenter starts stroking the participant's left hand with a paintbrush while doing the same simultaneously at the same position on the rubber hand with another paintbrush. This brings feeling where the participant has an illusion that the rubber hand was his own hand.

Embodiment has also been used in robotics for multiples purposes. The rubber hand illusion can also be achieved alone without experimenter through bilateral teleoperation Master-Slave robot system were the operator strokes the rubber hand with an haptic feedback enabled pen, and the robot side strokes simultaneously the participant's hand [Hara et al., 2016]. Embodiment has been shown to increase the acceptance of the robot by humans [Ventre-Dominey et al., 2019] and embodiment of a human on a robot has been proved to be preserved even through partial control of the robot (stationary participant but walking robot for example) [Aymerich-Franch et al., 2016]. Embodiment during a robot teleoperation task has been proved to lead to better user-satisfaction on the command and less time spent on the task [Almeida et al., 2014] and lead to illusory haptic feedback [Aymerich-Franch et al., 2017]. Embodiment has also been shown to be dependant of the time delay inherent to the robotic system [Arata et al., 2014] and crucially has also been shown to improve attention allotment, and hence performance in a dual motor task through an embodied robot system [Iwasaki et al., 2022].

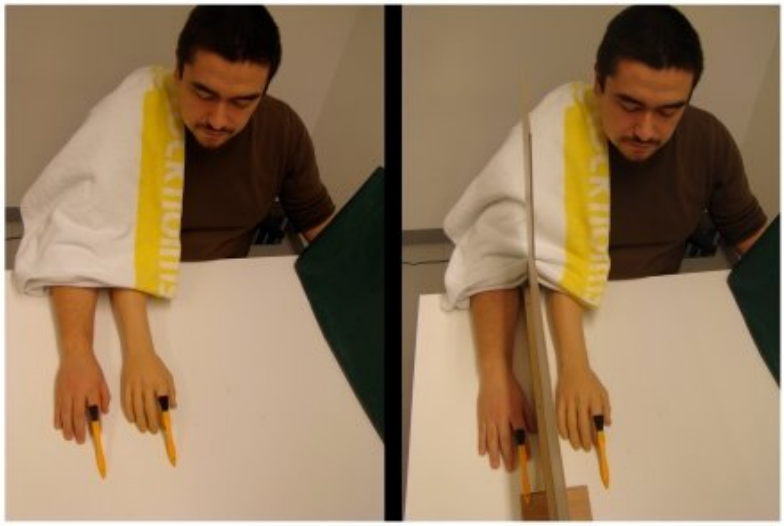


FIGURE 1.3 – *Rubber hand illusion set-up picture Source : [Guterstam et al., 2011]*

1.7 Embodied robot system

As discussed in the precedent section embodiment can produce several improvements during robot teleoperation. Hence in this thesis we will postulate that robot embodiment improves performances of human using a robotic system. There could be increases of concentration, reduce the feeling of scale inconsistencies, or of the robot DOF compared to the human. These same considerations have been taken in other works that uses embodied robots [Toet et al., 2020b, Iwasaki et al., 2022]

Research started since a few years ago to develop Embodied robot systems for multiple use-cases as wide as underwater robotics, UAV control, full body humanoid control and robot arm control. (Table 1.4). This research topic has been fueled by the emergencies of recent disasters like the 2011 earthquake followed by a tsunami that stroke the Fukushima region of Japan and its nuclear power plant. The radiation emitted in the region due to the incident affected the health of workers that helped rescue and decontamination, [Hiraoka et al., 2015]. Sending robots, if reliable and controlled in the right way could avoid these issues.

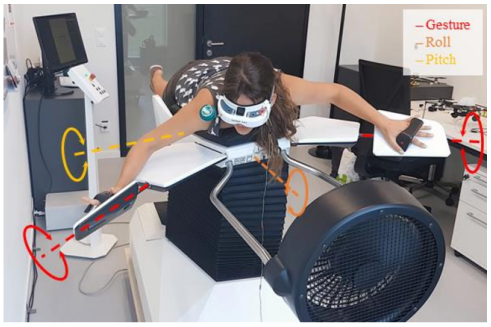
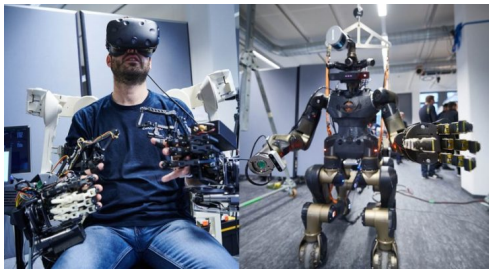
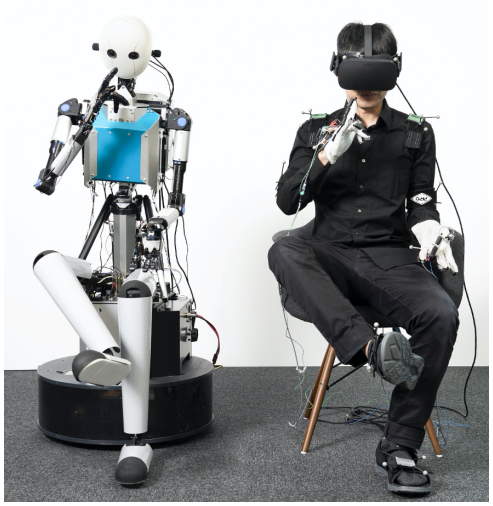
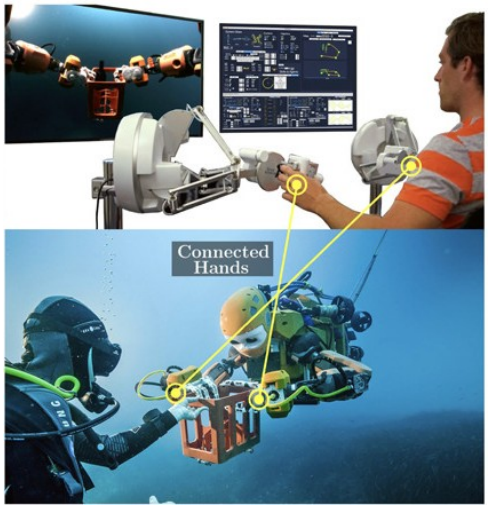
"Embodied Drone"from EPFL [Cherpillod et al., 2017]	"Centauro" [Klamt et al., 2019]
	
"TELESAR VI from Tachi Lab" [Tachi et al., 2020]	"Ocean One" [Khatib et al., 2016]
	

FIGURE 1.4 – *Embodied robot for UAV,UUV, Humanoid and Rescue Robot.*

Embodied robots are also starting to emerge in the manufacturing and service industry. They logically emerged at a faster pace than the research ones. Since 2020 and

the pandemic, we observed a lot of companies developing embodied robots whose main role is to interact with the environment. This might be explained by the surge of the remote work and tele-presence trend that emerged worldwide during the Covid19 related lockdowns. But due to the nature of under Research and Development work produced by companies, it is difficult to obtain information about their products. In 2020, when Telexistence (a Tokyo based startup) unveiled a video of their prototype,(Fig. 1.6) the upper limb humanoid robot is seen carrying plastic bottles to put them over their respective shelves in a convenience store while being remotely controlled with a VR HMD equipped operator. In 2021, Honda Research Institute released a video of a prototype doing precision prehension remotely controlled with a VR HMD equipped operator.(Fig. 1.5). And in 2022, West Japan Railway Co. released multiple footage of a prototype they plan to be ready by 2024. It consists in an upper limb humanoid robot attached to a construction vehicle in place of the nacelle (Fig. 1.7) The operator here remotely controls with a VR HMD the arm and the head rotation of the robot. They claim this product will be useful to repair railway pillars (Fig. 1.7).

We set up an embodied robot setup using haptic feedback, VR and 360° camera which is presented in Chapter 3 We did not proved that embodiment phenomenon occurred in our setup as it was not the main point of this PhD, but our setup was used in another research work and it was proved to create embodiment, [Iwasaki et al., 2022]. Therefore we will postulate that embodiment is occurring in the concerned experiment using our setup.

FIGURE 1.5 – *Honda Research Institute's Avatar Robot, credit: DPCcars*

FIGURE 1.6 – *Telexistence Robot operating in a convenient store, credit: Telexistence.inc*

FIGURE 1.7 – *Rei shiki Robot operating metallic parts over a railway, credit: The Japan Time*

1.8 Machine and Deep Learning

1.8.1 Motivation

Machine Learning techniques including Deep Learning are now popular in both research and companies since a few years. This can mainly be explained by two phenomena, the price reduction of extremely powerful GPU equipped computers and the availability of massive amount of data generally referred as big data. As storage became cheaper and internet access faster, more and more databases became available online from medical analysis results, categorized images, car driving pictures, music classifications, etc. The robotics field is also evolving using Deep Learning but robots impose specific constraints not present in other fields. Big databases of robot control, motion, sensors, etc. are difficult to create since it would require hundreds of hours of robot motion data. These constraints led to multiple robotic-related fields of research like Reinforcement Learning [Zhang and Mo, 2021], One-Shot Policy Learning [Finn et al., 2017, Santoro et al., 2016, Snell et al., 2017], Learning from Demonstration [Ravichandar et al., 2020].

That Deep Learning ia a popular technique, is not the reason we chose to use it. Its main advantage is its ability to correlates values, that at first sight did not seem correlated and to learn patterns without prior knowledge about the input data. Indeed these properties make Deep Learning interesting to learn from human motion and decision making. As we still don't fully understand human's complex decision taking process, this leads to situation in which it becomes hard to predict what will happen so it can be "hard-coded" so a robot can act in the same way. In such scenarios, Deep Learning becomes can learn and reproduce behavior that any human would naturally have, but that would be hard to describe in terms of robot behaviors.

1.8.2 Learning from Demonstration : Overview

Reinforcement Learning(RL) has been applied in robotics to multiple tasks like autonomous helicopter flight [Bagnell and Schneider, 2001],jumping dog robot [Kolter and Ng, 2011], cart pole swing [Deisenroth and Rasmussen, 2011], robot hand completing rubik's cube [OpenAI et al., 2019], the really impressive simulation of humanoid and wheeled robots in Isaac Gym [Makoviychuk et al., 2021] and the robotic assembly warehouse simulations [Narang et al., 2022]. However when using Reinforcement Learning, the trained agent is not taught by any human, but has learned by itself, unsupervised, in a simulator where the goal is to achieve a task through trial an error. Reinforcement Learning methods use a cost function as a reward in order to teach the agent if the trial was good enough or not, so it can gradually improve. While these methods are popular they have strong requirements:

- The need of a realistic simulator of the robot and its environment.
- A good knowledge of what task should be achieved in order to define the reward/cost function.

- Thousands of simulations that can take a really long time to train.

With these requirements, from a robotics point a view, a lot of task become impossible. For example every Human-Robot Interaction is hard to reproduce due to the difficulty to simulate human behavior, and it is not ofter possible to determine for every task where what should be done. This later point is important when working with humans. Indeed there are a lot of tasks where it is not exactly well known how the human does take its decision according to a lot of parameters.

An other method where learning is used to operate a robot is Learning from Demonstration (LfD) [Argall et al., 2009, Calinon, 2018]. The advances in the field were less impacting and seemed to go at a lower pace than the previously shown Reinforcement Learning based task. The reason of this lack of impact is due to the nature of Learning from Demonstrations : A human expert is needed to show the task to the agent.

The process of LfD is generally done through the following steps [Pais Ureche and Billard, 2015]
- Human expert achieves a demonstration in the task space and all data is recorded.
- This data is treated, filtered or analyzed in order to fit in the learning algorithm that will come next like creating windowed-time series, video from pictures or vice-versa.
- The agent is trained with the previously recorded data by using either Deep Learning algorithms, Policy Learning algorithms or more classical algorithms.

But most LfD methods have a fundamental problem : the body of the teacher and the robot's body are different [Dautenhahn and Nehaniv, 2002] (Chapter 14)
The kinesthetic method, that consists to have the user move the robot parts while it is in gravity compensation mode, brings a nice aspect due to the fact the human can feel applied forces trough the robot, feel mechanical constraints, etc. But this method has a drawback when a high number of DoF should be moved by the human, with humanoid for example it is difficult to imagine moving the whole robot like this while still using the temporality contained in the data. Same goes to some extent to 7DoF robots, since the redundancy will have to be managed.
Moreover in case of the need of visual data, the point of view of the human and the potential camera field of view are likely to be different because it is not acceptable for the human to obstruct the camera's field of view. The human will teach from their own point of view, but it might render differently from the robot's one.

As explained in [Ravichandar et al., 2020] there are three main type of demonstrations Teleoperation, Observation and Kinesthetic (Fig. 1.8). They can be classified in three classes : Easiness of demonstration, possibility to control a high number of DoF and easiness of mapping teaching and actions to realize.

We trust that Embodied Teleoperation could have the same advantages regarding LfD, as the Teleoperation row in the table (Fig. 1.8) but it would keep the Ease of Demonstration as explained in the sections 1.7 and 1.6

Demonstration	Ease of Demonstration	High DOF	Ease of Mapping
Kinesthetic	✓		✓
Teleoperation		✓	✓
Observation	✓	✓	

FIGURE 1.8 – *Differences between LfD teaching methods and their respective advantages Source : [Ravichandar et al., 2020]*

Our concern about the expert demonstration is also explained in [Sasaki and Yamashina, 2021] : *Unfortunately, it is often difficult to obtain optimal demonstrations for many tasks in real-world problems because the expert who tries to operate the robot so that it can achieve tasks often makes mistakes due to various reasons, such as the difficulty of the task, difficulty in handling the controller, limited observability of the environment, or the presence of distraction.* Hence we propose an embodied teaching methods using haptic feedback teleoperated robot enhanced with a VR HMD device. We used short video clip of a robot controlled by a human expert to train our agent composed of a pre-trained ResidualNetwork (ResNet) and Gated Recurrent Units (GRU). The details of the model and methods used to train our agent will be detailed in Chapter 6.

1.8.3 Differences with learning in video games

One could think that many complex problems in deep learning are already solved. Computer graphic fields, video games and robotic, have a lot in common in terms of simulation. They all use Inverse Kinematics to generate motion, body dynamics, etc... however they differ in the sense that they can simulate everything and get huge amount of data by letting a simulator run one week to simulate 10 years of time. The key difference from our work is that in our study we will try to learn human behavior, and for this, as will be discussed in the later chapter, embodied teleoperation provides some key advantages.

Indeed in 2015, Alpha Go (a specialized Deep Learning Based Agent) beat the Korean world Go champion. The most arcade video games are currently played as better by AI as humans thanks for example to the Open AI Gym framework [Nichol et al., 2018]. More recently, Real Time Strategy video games like Starcraft, which consists in planning challenges of multiple hundredth of units over hours of time in order to beat the opponent player, has been participant to a newly developed network called AlphaStar which has reached the Grandmaster level by using supervised learning from players data and simulation of more than 200 years (per different playable classes) with reinforcement learning [Vinyals et al., 2019]

Racing games are also concerned by the recent advances of the technology, [Weiss and Behl, 2020] presented a control technology using three high-level approaches : mapping images, trajectory Waypoint Prediction and Trajectory Prediction with Bézier Curves Trajectory Prediction to control actions like steering by using supervised learning from players data with haptic steering wheel controllers. This researches are also clearly linked to autonomous driving.

But these techniques also led to robot skill learning as it can be seen in NVIDIA's ISAAC Gym Reinforcement Learning simulator [Narang et al., 2022] (Fig. 1.9). Some of those skills can be applied on real robot thanks to transfer learning [Kim et al., 2022].

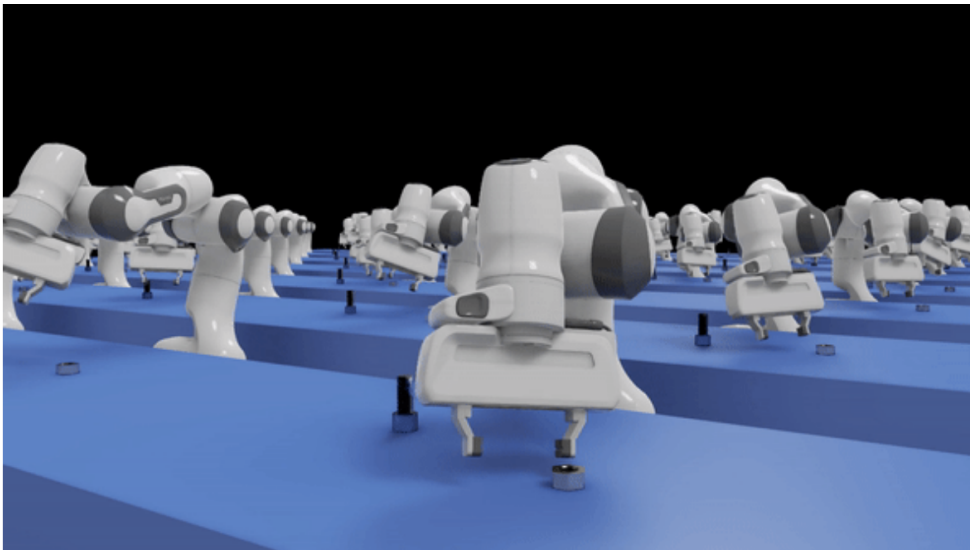


FIGURE 1.9 – *In parallel,128 Franka Panda Robot are tightening nuts on bolts inside the NVIDIA Isaac simulator Source [ISA]*

1.8.4 Learning from Observation

Since the recent success of deep learning for learning features of complex visual images input, vision-based control has become more considered in the field since it has the interest to learn to control dynamical systems directly from raw images input. These methods that learn policies to map visual input to classified movement action were used to teach an agent to play Atari Games [Mnih et al., 2013] and control predefined motion of a manipulator robot [Watter et al., 2015] , inverted pendulum in simulation [Wahlström et al., 2015] [Levine et al., 2015] or even the billiard physics [Fragkiadaki et al., 2015].

After the use of visual-based control in robotics, some works proposed methods to learn Physics from Images stream input, ie. videos input. [Wu et al., 2015] and also from static images [Mottaghi et al., 2015] It means it also become possible to control, or predict physical motion which is useful in robotics or human motion field by learning dynamics of motion, collisions, etc.. For example, in [Jung et al., 2018, Rodríguez-Hernandez et al., 2019] the authors use a CNN architecture to train an UAV to fly through a gate by using visual data generated by an expert pilot.

CHAPTER 2

CONTROL OF AN EMBODIED ROBOT : THE CO-LIMBS CASE

Contents

2.1 Introduction

The promising possibilities offered by supernumerary robotic wearable arms are limited by the lack of an intuitive and robust user interface to control them. Here, utilizing admittance control, we propose a 'Collaborative limbs', or 'Co-limbs' user interface for wearable robot arms. The key feature of this user interface is its intuitiveness enabling even first time users to immediately move and use the normally stiff robot arms for assistive tasks and even teach the robot simple and useful movements. We demonstrate the diverse range of applications enabled by this simple yet powerful user interface through example demonstrations in the *Passive Assist*, *Power Assist* and *Playback* modes.

Imagine if you had two extra robotic arms to enable you to pull your luggage in the airport and hold your coffee cup while you take the hand of your child, or to assist you in holding a heavy box, or even to wave a hand fan back and forth on a hot day while you read a newspaper. These promising and exciting possibilities have led to the development of supernumerary Wearable Robotic Arms (WRA) by several groups [Llorens-Bonilla et al., 2012, NAKABAYASHI et al., 2017], including ours [Sasaki et al., 2017]. Yet, these designs remain limited to demonstrations, and have not found commercial popularity because of the lack of intuitive and efficient user interfaces to operate them.

The Co-Limbs user interface uses force sensors and admittance control to enable a user to intuitively move and use the otherwise non-backdrivable wearable robotic arms.

A good user interface should be intuitive to use and should enable the user to easily convey to the robot what he/she wants, and how to achieve it. The first requirement for any operation with the robot arm (like holding a cup, pulling the luggage, etc.) is for it to be positioned and oriented appropriately. However, even this first operation is difficult in the current WRA because of the popular use of servo motors in these wearable systems- which helps reduce their weight, but the high gear ratios in these motors results in the robot being stiff and servo control makes the robot non-backdrivable. Positioning the robot arms thus requires the use of either joysticks, or tracking of other limbs, which, due to positioning being in 12 dimensional space (3 position + 3 orientations) $\times$ 2 arms) and over a large workspace around the user's body, can require significant visual attention and time. To address this issue, here we propose a simple but promising user interface, called 'Co-limbs', that is based on the idea of *collaborative* control [Peshkin et al., 2001] in which the user and robot work on the task together. This interface is intuitive to use enabling even a first time user to immediately position, orient and use the device in multiple scenarios, and even teach the WRA simple movements.

2.2 Co-limb interface description

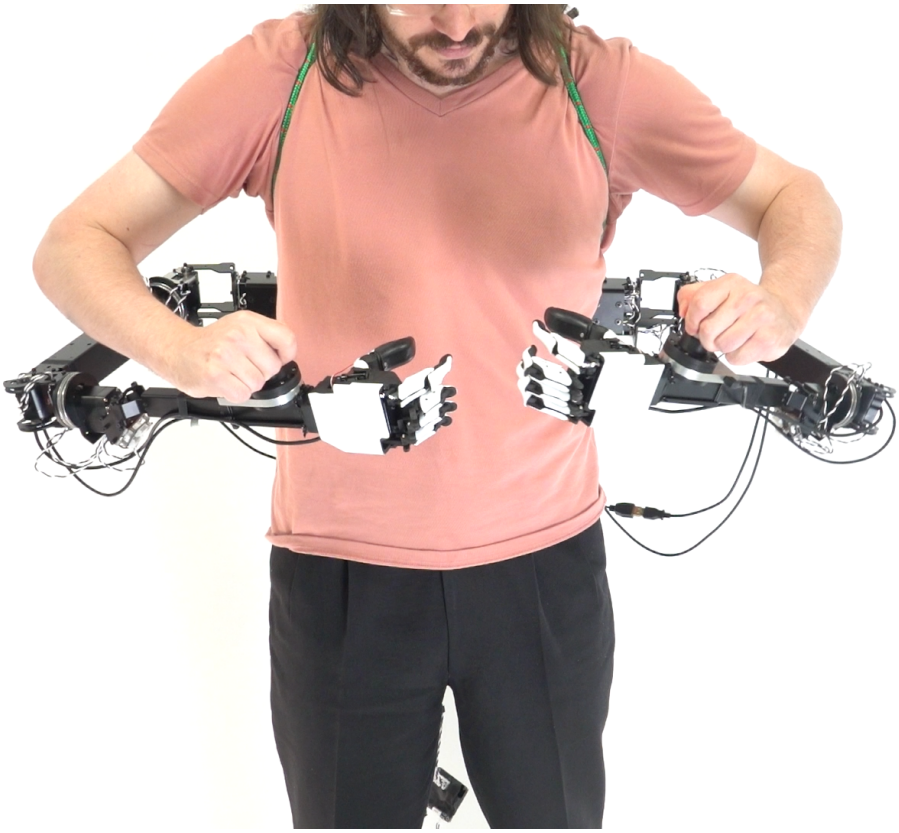


FIGURE 2.1 – *The Co-Limbs user interface uses force sensors and admittance control to enable a user to intuitively move through force sensors levers and use the otherwise non-backdrivable wearable robotic arms.*

Humans are adept in embodying and using hand-held tools [Ganesh et al., 2014, Tee et al., 2018]. They can immediately position and orient a hand held tool as per task requirements, often requiring minimal visual feedback in order to do so. We therefore hypothesized that an interface that promotes the wearable robot arms to be viewed as tools by the user, would automatically make it easy for the user to operate them. This however, is not directly possible in non-backdrivable system, like servo-motors. But servos have the advantage to be light and do not require any external Pulse Width Modulation (PWM) controller or mechanical brake. This can save a lot of weight in the prototype of a Wearable Robot Arms

Here we used a Wearable Robot Arms prototype called Meta-Limb that was developed by Karakuri Product and Tokyo University's Inami living lab [Sasaki et al., 2017, Maekawa et al., 2019]

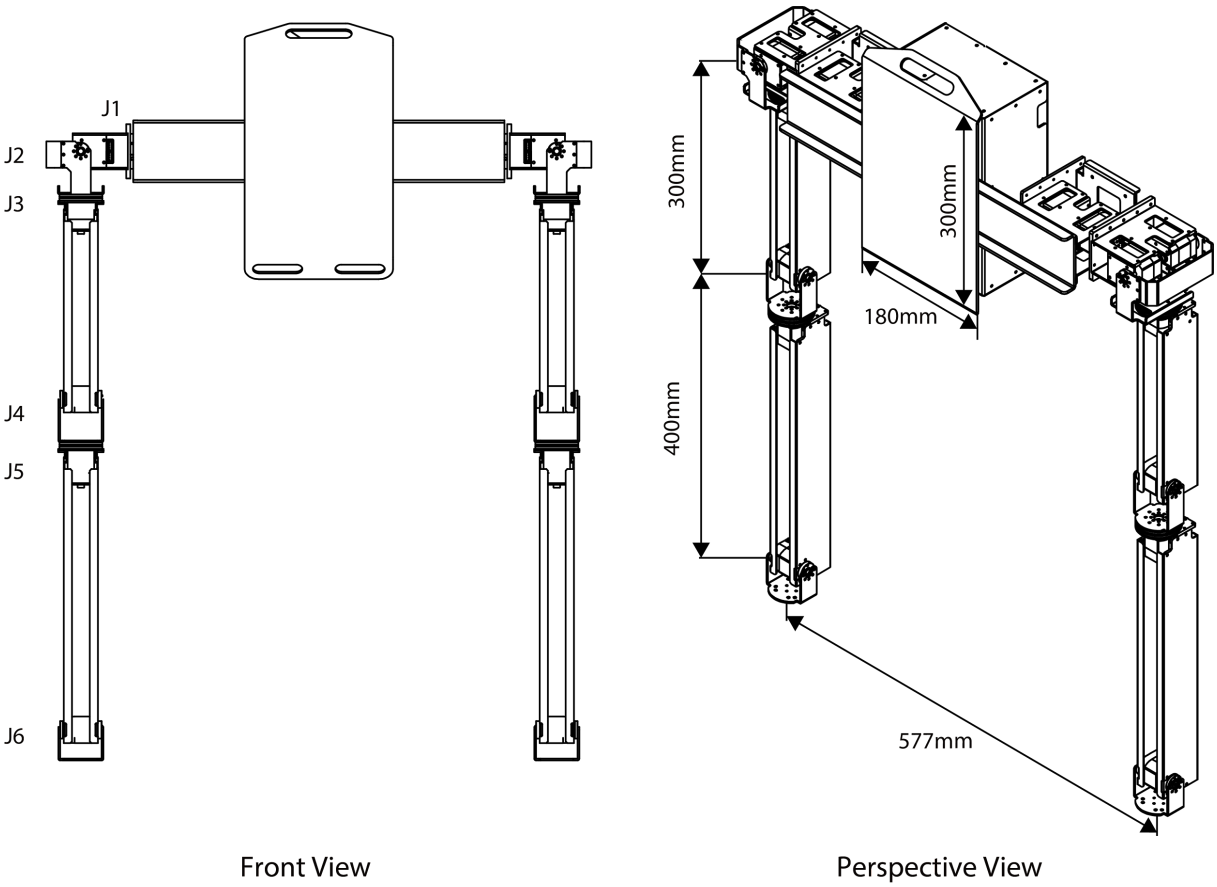


FIGURE 2.2 – *Schematic of the frame of Meta-Limb, developed by Karakuri Product and Tokyo University's Inami living lab*

Our work uses the aforementioned prototype that has been used for multiple works. [Sasaki et al., 2017, Maekawa et al., 2019, Izumihara et al., 2019] Each arm is a 6DoF robot arm that was designed to match human morphology. But in this previous works,

Meta-Limb was controlled with Unity, a 3D graphic engine used for Video Games and VR, using Inverse kinematics and rigid bodies in the simulation. This choice is backed by the fact that this prototype was used to be teleoperated by a remote user with a HMD headset and a pair or controllers. To match our expectancies in term of robotics, we decided to change the control software to OpenPHRI, [Navarro et al., 2018], an open source robot control framework, in order to achieve a co-bot like control. Thus Co-Limb is the name of this collaborative prototype based on Meta-Limb.

The Co-limbs system thus yields to have a handle equipped with a force sensor (Fig. 2.3) (two Leptrino 6 Axis Force Sensor 055YA 501) on the forearm of each WRA and utilize *admittance control* to allow the user to move the non-backdrivable arms.

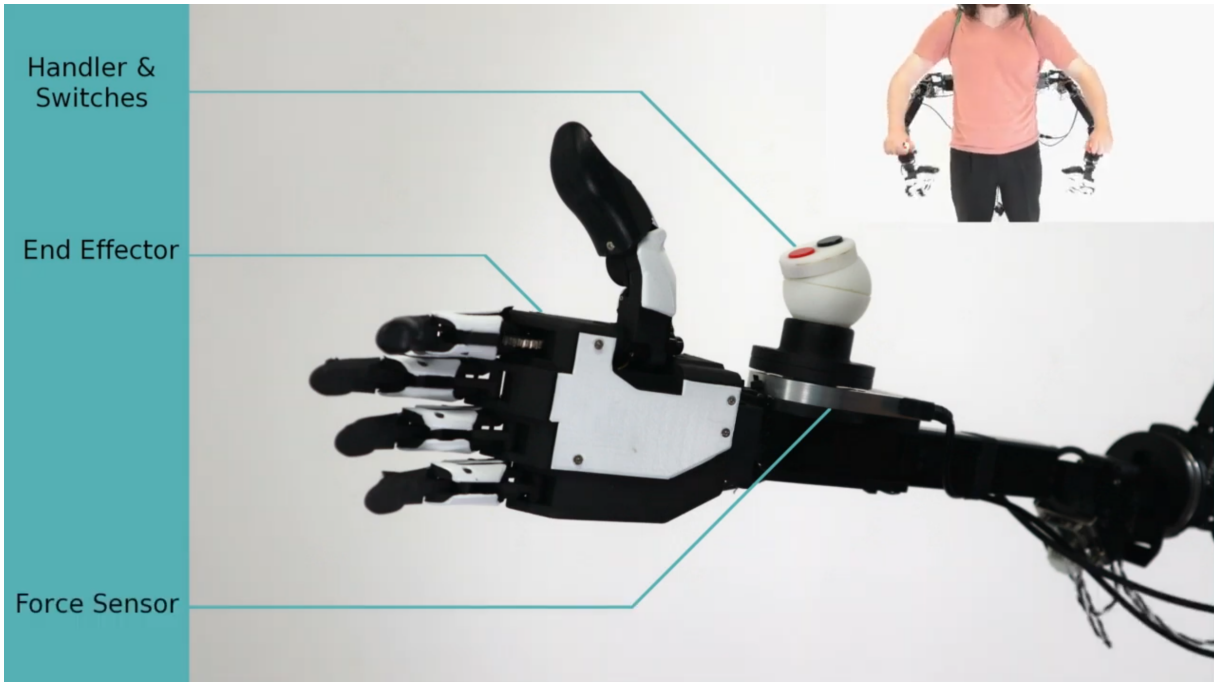


FIGURE 2.3 – *Co-Limb's end effector with the switches present on each handles that lets the user clutch the hand and trigger motion recording/replay for one arm.*

In admittance control, the torque *wrench* W applied on each handle is recorded in the force sensor coordinates ($W_{fsensor}$), and, knowing the current pose of the robot arm, transformed in the world space coordinates ($T^{world}(Q)_{fsensor}$), with Q a 7-dimensional vector of the current joint states of each arm The force is then converted to a corresponding velocity in Cartesian coordinates with an assumed dynamics $\omega_{world} = \delta(m, d, W_{world})$, with mass m and damping d. Finally the joint velocities is calculated using inverse kinematics as $\omega_{world} = IK(\omega_{world}, Q)$. Our current application is implemented in Linux using $m = 1$ kg, $d = 15$ kg/m and the inverse kinematics functionality provided by the openPHRI package developed in our lab.

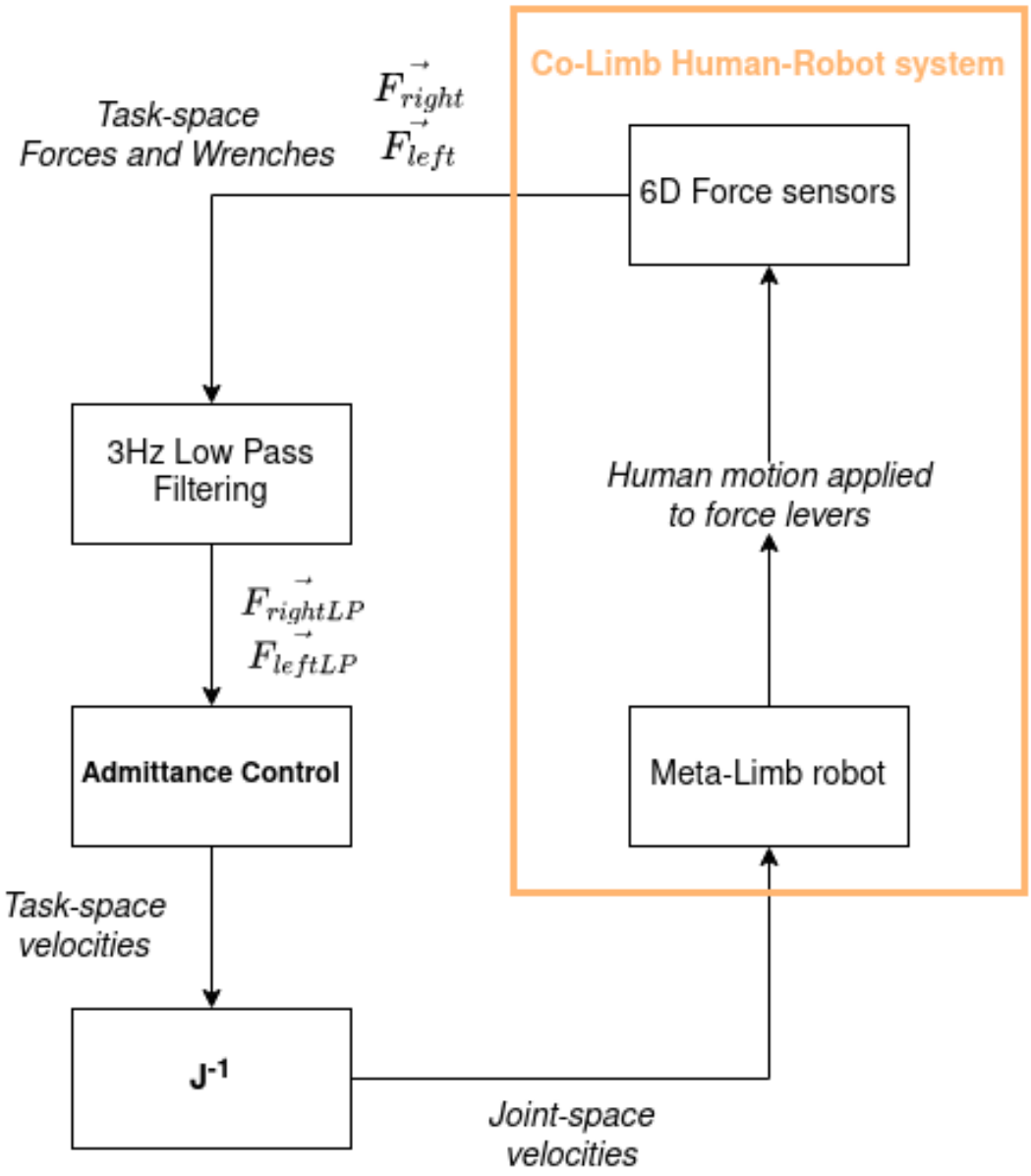


FIGURE 2.4 – *Admittance control scheme used for Co-limb*

2.3 Main features

Our interface has several salient features that radically expand the applications of a WRA systems.

- Its intuitiveness, enabling even a first time user to use it immediately with no training (Fig. 2.5)
- Back-drivability, a key feature for user comfort, and enabling the user to easily position the device for applications of *passive assistive*.
- Collaborative guidance: Two fundamental challenges for a robot interacting with a human is to (1) understand the intention of the user, and (2) to plan its own mo-

vements to help the user accordingly. Our interface proposes to overcome these issues by allowing the user to collaboratively guide the robot, while benefiting from the *power assistance* [Lee et al., 2012].

- Better teaching:The collaborative guidance opens up the possibility of using *teaching by demonstration* [Haage et al., 2017] techniques to improve the skills of the WRA (c.f. demonstration video).

- Versatility: While we do not show this in our current demonstration, the user interface also allows us to modulate the dynamics felt by the user, allowing us to make the WRA feel heavy or light, use the robot system to cancel possible tremors and noise in the user input (for example when the system is used by a elderly user)

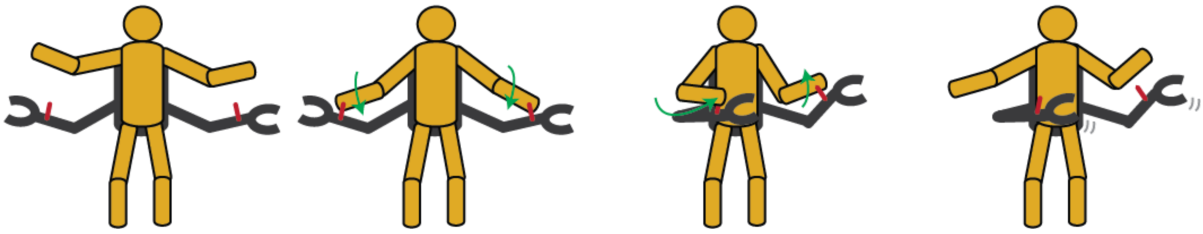


FIGURE 2.5 – *Instruction to use Co-limb from Left to Right : First the user should grab each force handle, then apply some force to have the robot move. If the record switch was pressed while the force was applied, the motion can be replayed without having to apply forces*

2.4 Possible applications

We demonstrate the intuitiveness and versatility enabled by our proposed user interface in three example modes of application utilizing the WRA device [Sasaki et al., 2017].
First, the *Passive Assist* mode allows the user to orient the robot hands in the desired postures and to utilize them in given scenarios (e.g. pull a suitcase or hold an umbrella, c.f. video). The 1 DoF robot hand (open or close fingers) is activated by a switch on the handle.
Second, the *Power Assist* (Fig. 2.9) mode allows the user to guide the robot arms and pick up cumbersome loads (e.g. lifting a big box, c.f. video). Note that the weight of the load is obviously transferred to the user (as the robot system is worn by the user), but the robot distributes the load over the back and waist making it more comfortable than lifting the load with their hands. The Power Assist mode can be particularly useful for elderly users as it can assist their own body weight, for example, assist them in standing up as in the third picture of the second row of (Fig. 2.6)
Third, the *Playback* mode allows the users to record and playback simple but useful repetitive movements to the robot, so that the robot can then perform them without user guidance (e.g. use a hand held fan, c.f. video). The record process was done by recording the end effector trajectory and replaying it afterward. (Fig. 2.6)

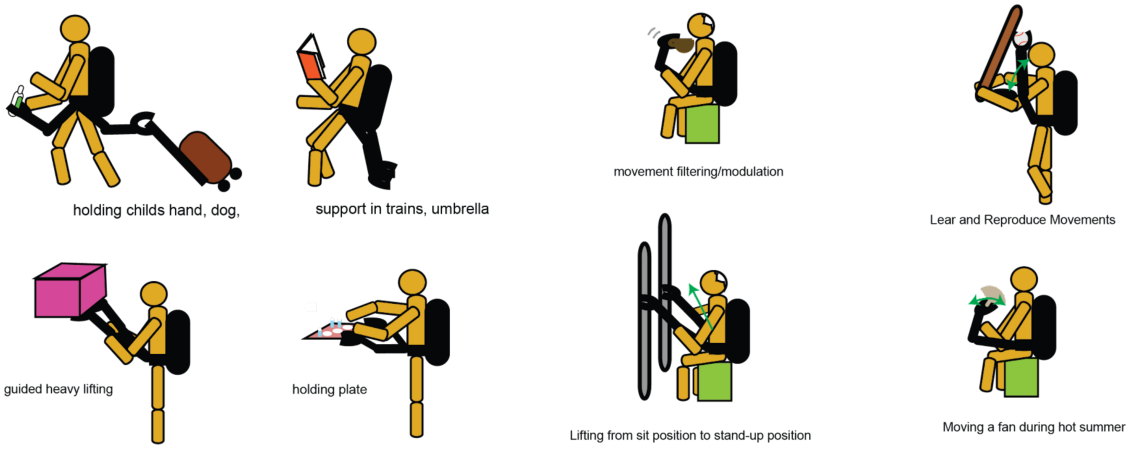


FIGURE 2.6 – *Examples of possible passive and active applications*

FIGURE 2.7 – *Youtube Video of our Co-limbs Experiment (the QR code is a also clickable link)*

FIGURE 2.8 – *Examples of application : record a motion to have a fan move*

FIGURE 2.9 – *Examples of application : Power assist*

2.5 Contribution trough our Co-limb interface

In this study, we introduce a simple user interface for supernumerary wearable robot arm systems. This interface utilizes a force sensor to enable users to guide the robot arms and enable various passive and active assistance tasks, and enabling the user to teach simple movements to the robot. The key feature of this user interface is its intuitiveness and ease of use. We currently demonstrate the versatility of the interface in three modes of application, and we are now developing its applications in the field of elderly care where we believe it can be extremely useful.

2.6 Presentation at ACM Siggraph Asia

Our work has been presented at the Siggraph Asia 2019 conference in Brisbane in the emerging technology paper. Siggraph conferences have a particularity for emerging technology papers : the authors must bring their prototype to the conference and have it displayed and tested by the public. This led us to spend three days displaying Co-Limbs prototype (Fig. 2.10) to the public. We gathered interesting feedback from the public, that seemed very happy to try Wearable Robot Arms for their first time (Fig. 2.11). We had set-up multiple objects that they could grasp with the robot like balls, plastic tray and sponges. But we also had some non anticipated problems in the prototype electronics, and despite the fact we had brought spare components, it led in having one arm not functioning during the last day. Despite all the fix we had to do on Co-limbs during the conference, where it was used all day by the public, the safety of the public was maintained and we could keep it almost functional until the end.

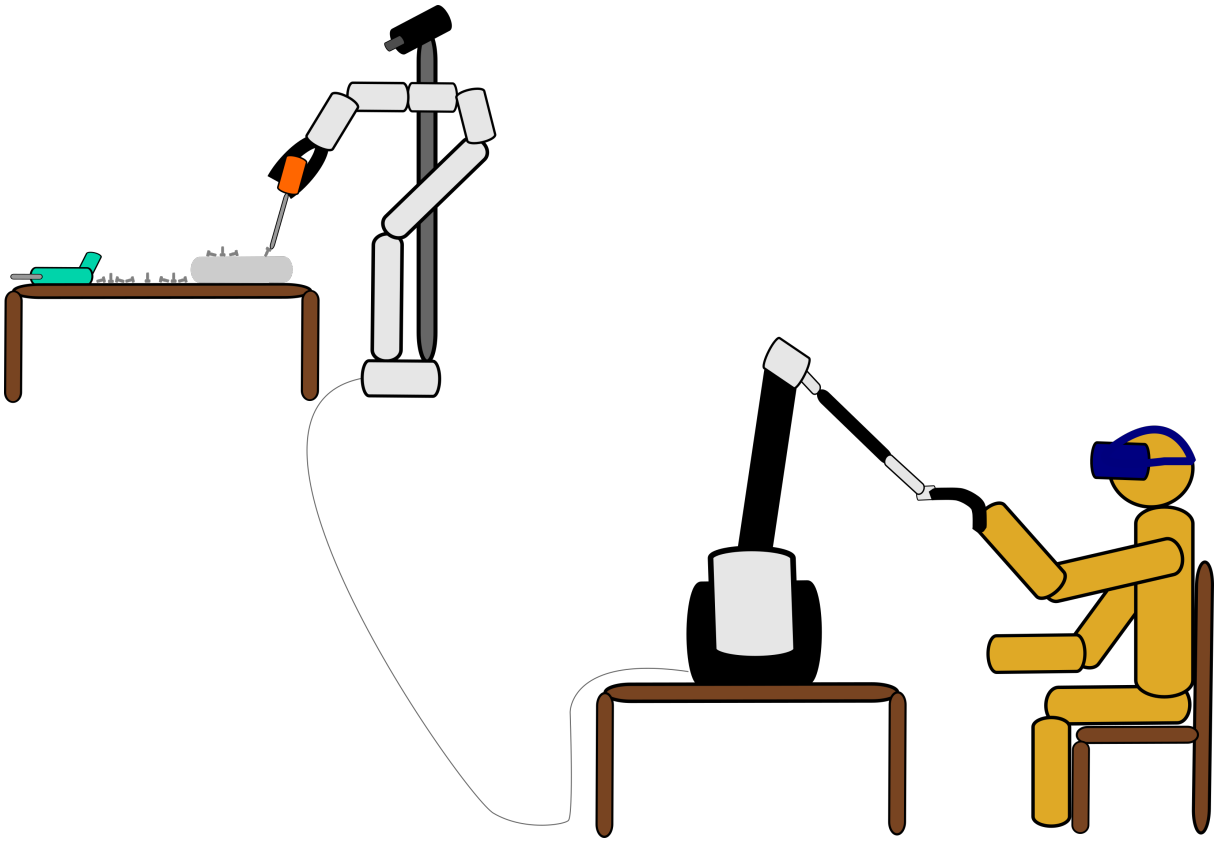
FIGURE 2.10 – *Co-Limbs in our booth at Siggraph Asia 2019*

FIGURE 2.11 – *Co-limbs being tested by the public during Siggraph Asia 2019*

CHAPTER 3

EMBODIED ROBOT SETUP AND CONTROL

Contents



3.1 Introduction

In Chapter 2 we used the Wearable Robot Arm system of Tokyo University to induce the sense of embodiment to the user. In this chapter we will explain the technical choices we took when building our embodied robot system.

In order to reproduce the sense of embodiment which is needed in our teleoperation experiments, we spent a non negligible time setting-up the Experimental Setup. The key goal of the setup is the induce a feeling in the human user controlling a robot arm that he is controlling his own arm. The setup is made out of 4 principles devices :

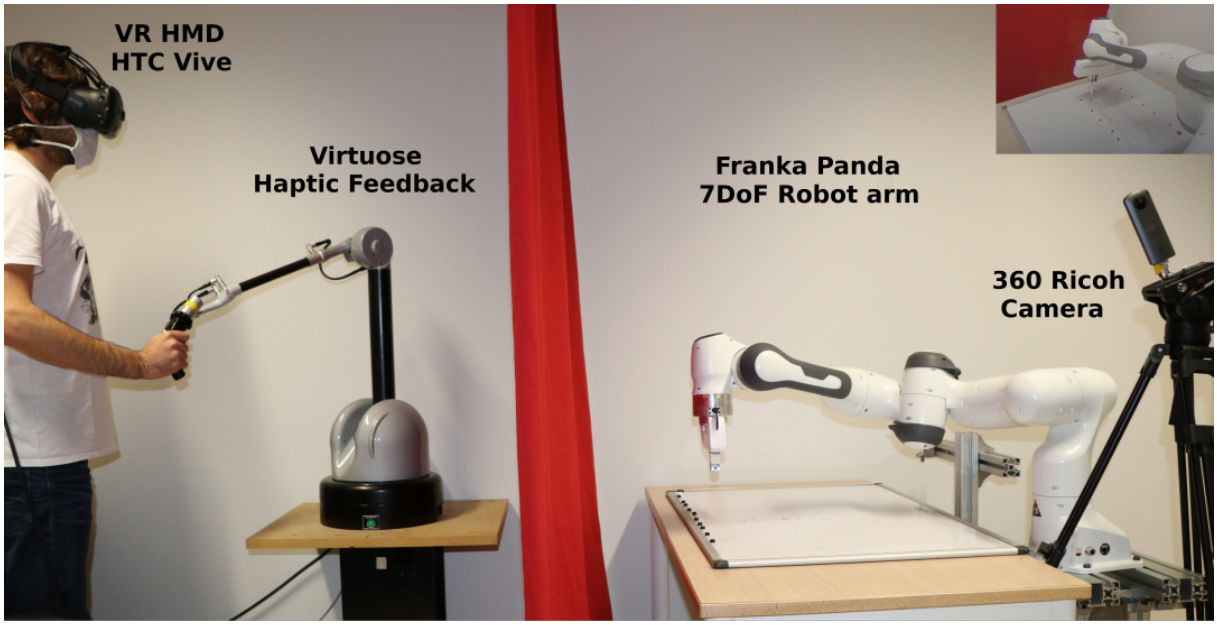


FIGURE 3.1 – *Embodied teleoperation setup. Our setup consists of a Virtuose haptic device and a Franka Panda robot. The operator wears a head mounted display and operates the robot using the haptic device. He is provided with a first person visual display from a camera placed above the Franka. The controller utilized to help him guide the trajectory and impedance of the robot is explained in Sect. 3.4.*

3.2 Hardware setup

- Haption Virtuose 3D large workspace haptic feedback device [https://www.haption.com/fr/products-fr/virtuose-3d fr.html]
- a serial 7 DOF Franka Panda robot manipulator [https://www.franka.de/technology]
- A VR Head Mounted Device HMD [https://www.vive.com/fr/product/vive pro/]
- A pan-tilt mounted stereo camera / a 360° camera

3.2.1 Haptic Feedback Device

To completely feel the sense of embodiment, the user also needs to be able to feel its own environment. This will be achieved trough an haptic feedback device that let the user feel forces on the top of letting the user move freely the pose of the Haptic Feedback device's TCP in space. Our Haptic feedback device is a Haption Virtuose 3D Fig 3.2 which can feedback a 3D linear force up to 10N constantly and 35N at peak. Torque feedback were not possible with this model. We chose to control our haptic feedback device in admittance mode, which means the input of the control of the haptic feedback device is a force and the output a position. It fits the need of the robot being controlled in impedance in the experiment we will explain in the chapter Sect. 4

FIGURE 3.2 – *The Haption Virtuose 3D, our Haptic Feedback device, fixed to a high impedance base. Indeed for the need of our vibration-based impedance estimation method, we opted for this base in order not to estimate the furniture it's placed on impedance instead of the human's arm one.*

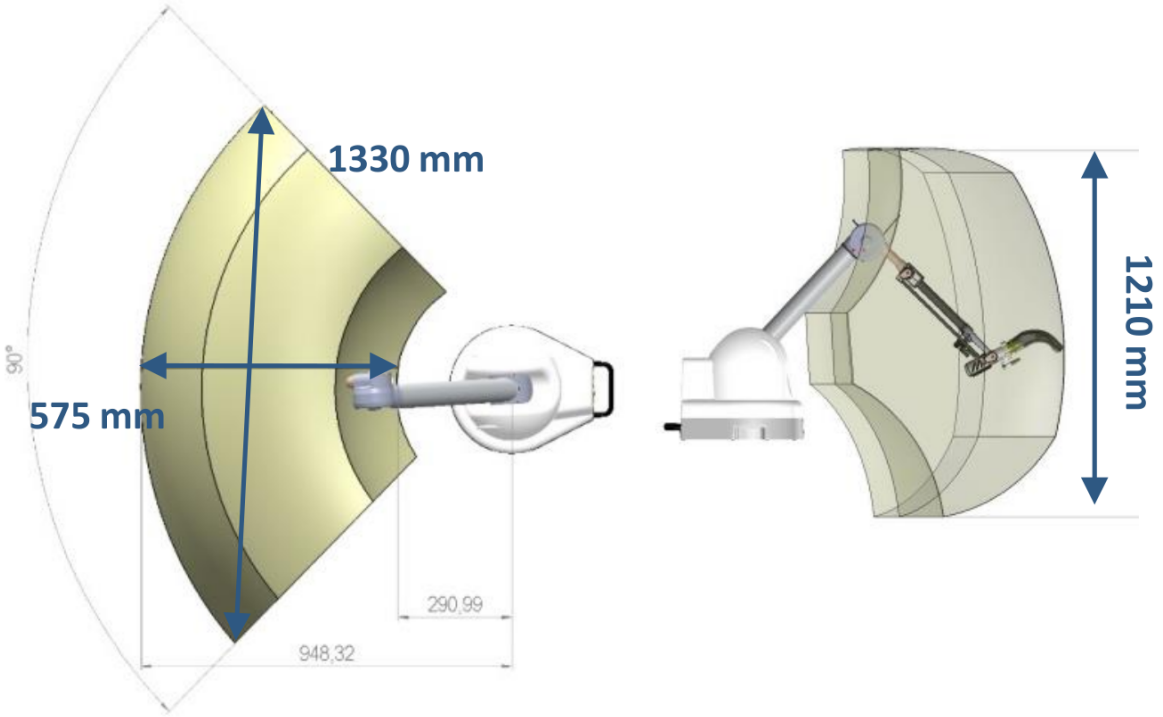


FIGURE 3.3 – *Virtuose's workspace, Source : [https://www.haption.com/fr/products-fr/virtuose-3d fr.html]*

3.2.2 Robot manipulator

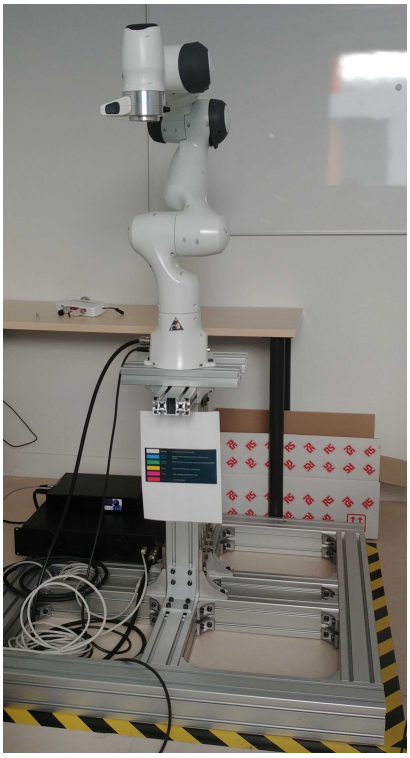


FIGURE 3.4 – *The 7DoF Franka robot fixed to its base. This aluminium profile made base provides a sturdy base on which the robot's TCP can reach positions below it's base*

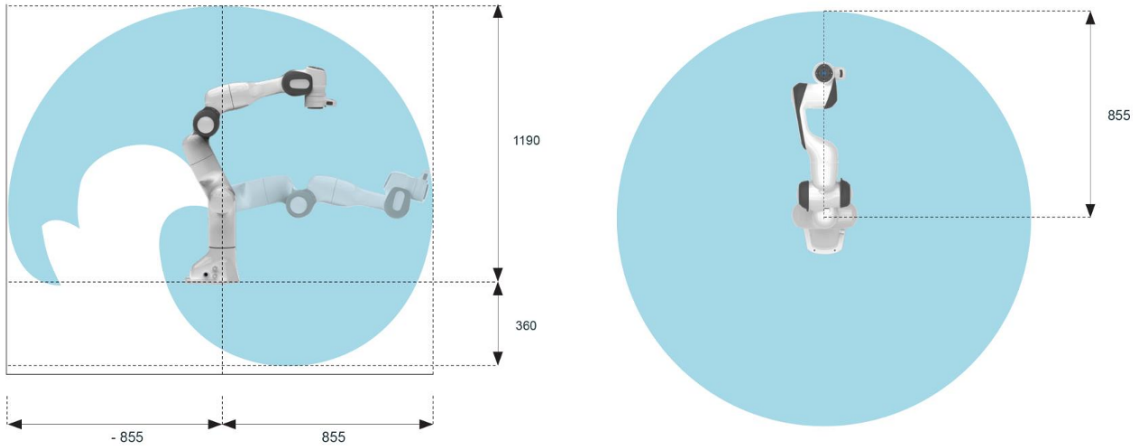


FIGURE 3.5 – *Franka's workspace, unit: milimeters , Source : [https://www.franka.de/technology]*

For the robot and haptic system we used a 7DoF serial manipulator Franka Panda Fig 3.4

The system is then set up to move the target of the manipulator according to the pose and velocity applied by the user on the force feedback device. Any force applied on the robot generated by an external physical contact would lead to a force feedback to the user. The controller used is detailed in Sect. 3.4

3.2.3 Virtual Reality Head Mounted Display

To achieve the sense of self-location near the robot, there is the need to transfer the viewpoint of the user near the robot position. In order to achieve it, we placed a pan-tilt mounted stereo camera near the robot first joint, at a distance comparable to the one an adult would have between his/her head and shoulder. The camera being a stereo camera it becomes possible for the user to have a sense of depth while watching the video stream. But in order to make use of the depth, we decided to send both video streams to each eye of a Virtual Reality Head Mounted Display (VR HMD) [https://www.vive.com/fr/product/vive pro/].

3.2.4 Camera setup

For our first experiment we used a consumer grade 360° camera, that would avoid the use of a pan-tilt device, by rotating inside the 360° spherical image instead of moving the camera. This led to smoother images since there would have been no movements of the lenses and thus the electronic image stabilisation is not used, and also less mechanical devices in the system to control with potential latency. We had planned to use two 360° cameras, but it seems that the stitch/un-stitch phase of two different 4K

streams at the same time was not possible on the computer with these cameras. Unfortunately, the 360 camera's we decided to use happened to have too strong delays (between 500ms and 1000ms) for the task during real time streaming. We think this is due to the image compression/decompression and stitching/un-stitching phases in the streaming protocol and the fact that these cameras used USB2.0 standard, which forces compression to a higher level than USB3.0 standard.

3.3 Software and middleware framework

Our C++ software programs were developed, using the PID11 (Packages Integral Development) API maintained by R. Passama and B. Navarro. The objective is to create en environment trough CMake where the user can program more easily by integrating tools that are recurrently needed in robotics applications like loggers, real-time plots, real-time web interfaces, different kind of filters, etc. It also manages dependencies through Linux automatically and let the programs be installed as standalone app, so that applications can be instantly deployed by any other user.

We also integrated ROS1 (Robotic Operating System) in our system in order to manage all the devices used in our embodied setup in a Linux computer. Indeed, ROS lets the user create nodes to integrate all the devices in the same network so they can communicate and send messages to each node. In our work ROS was used to have the feedback device communicate with the robot (Fig. 3.6)

The VR system was managed by another computer that managed Steam VR for the connection of the HMD to the system and Unity 3D to render the camera view in the VR HMD.

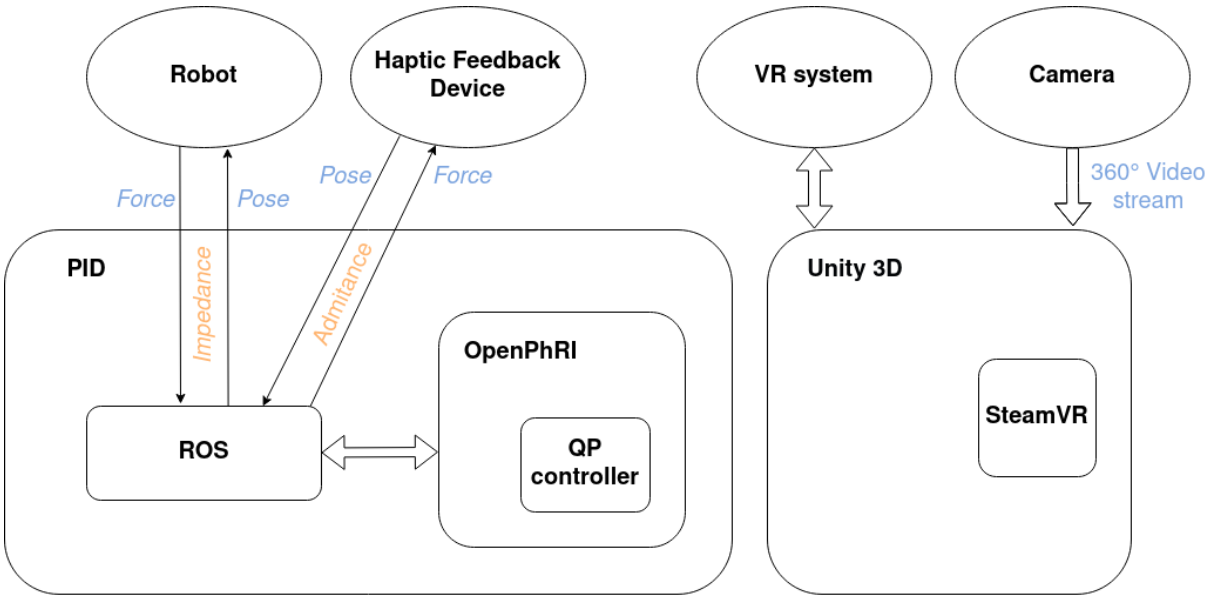


FIGURE 3.6 – *Global scheme of framework and software used for our setup*

3.4 Robot control

We consider a serial robot with k degrees of freedom with the following dynamic model:

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau}^* + \boldsymbol{\tau}_{\text{ext}} \tag{3.1}$$

$$\boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{\tau}_{\text{dyn}} = \boldsymbol{\tau}^* + \boldsymbol{\tau}_{\text{ext}}. \tag{3.2}$$

In these equations, $\boldsymbol{H}(\boldsymbol{q}) \in \mathbb{R}^{k\times k}$ is the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \in \mathbb{R}^k$ embeds Coriolis and centrifugal effects, $\boldsymbol{g}(\boldsymbol{q}) \in \mathbb{R}^k$ are the joint torques induced by gravity, $\boldsymbol{\tau}^* \in \mathbb{R}^k$ is the torque command and $\boldsymbol{\tau}_{\text{ext}} \in \mathbb{R}^k$ the external torques applied to the robot. $\boldsymbol{\tau}_{\text{dyn}}$ are the torques induced by Coriolis, centrifugal and gravitational forces

Since the goal of the controller is to realize Cartesian forces at the robot's end-effector, one simple way to compute $\boldsymbol{\tau}^*$ would be to use:

$$\boldsymbol{\tau}^* = \boldsymbol{J}_R^\top \boldsymbol{F}^{\boldsymbol{R}^*} + \boldsymbol{\tau}_{\text{dyn}} \tag{3.3}$$

where $\boldsymbol{J}_R \in \mathbb{R}^{6\times k}$ is the Jacobian matrix associated with the end-effector and $\boldsymbol{F}^{\boldsymbol{R}^*} = \boldsymbol{F}^{\boldsymbol{R}} + \boldsymbol{F}^{\boldsymbol{O}} \in \mathbb{R}^6$ is the force to be realized. This is made up of two components.

The first component is our impedance controller:

$$\boldsymbol{F}^{\boldsymbol{R}} = \boldsymbol{\alpha}\boldsymbol{K}\Delta\boldsymbol{x}^{\boldsymbol{R}} + \boldsymbol{\beta}\boldsymbol{D}\Delta\dot{\boldsymbol{x}}^R + \mathbf{F}_{\mathbf{I}}. \tag{3.4}$$

While the R subscript represents the robot, $\boldsymbol{F}^{\boldsymbol{R}}$ is the command force of the robot and $\boldsymbol{x}^{\boldsymbol{R}}$, $\dot{\boldsymbol{x}}^R$ are the movement and velocity of the robot relative to the reference from the operator (respectively x_r^O and $\dot{x}_r^O$); $\boldsymbol{\alpha}$ is a scaling parameter on the human stiffness and $\boldsymbol{\beta}$ a scaling parameter on the human damping. $\mathbf{F}_{\mathbf{I}} = m^R\ddot{x}^R$ represents an approximated Cartesian inertia compensation term.

The second component, $\boldsymbol{F}^{\boldsymbol{O}}$ is a pseudo interaction force that is used in Section Sect. 4.4 to simulate force perturbations from an assisted patient as explained in the next Chapter.

However, this approach does not ensure that the robot mechanical limits are respected. To cope with this issue, we use a quadratic programming approach including the joint position, velocity and torque limits, to ensure admissibility of the torque control inputs. The problem is formulated as follows:

$$\begin{aligned} \underset{\boldsymbol{\tau},\,\ddot{\boldsymbol{q}}}{\text{minimize}} \quad & ||\boldsymbol{\tau} - \boldsymbol{J}_R^\top \boldsymbol{F}^{\boldsymbol{R}^*}||_2^2 \\ \text{subject to} \quad & \boldsymbol{\tau} = \boldsymbol{H}\ddot{\boldsymbol{q}}, \\ & \boldsymbol{\tau}'_{\text{min}} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}'_{\text{max}}, \\ & \ddot{\boldsymbol{q}}_{\text{min}} \leq \ddot{\boldsymbol{q}} \leq \ddot{\boldsymbol{q}}_{\text{max}}. \end{aligned} \tag{3.5}$$

In this equation: $\ddot{\boldsymbol{q}}_{\text{min}}$ and $\ddot{\boldsymbol{q}}_{\text{max}}$ are $k \times 1$ vectors computed to include also the joint position and velocity limits (as in [Bouyarmane et al., 2017]), $\boldsymbol{\tau}'_{\text{min}}$ and $\boldsymbol{\tau}'_{\text{max}}$ are $k \times 1$

vectors accounting for the joint torque mechanical limits $\left[-\boldsymbol{\tau}_{\text{max}}\ \boldsymbol{\tau}_{\text{max}}\right]$ with the Coriolis, centrifugal, gravity and external torques removed:

$$\boldsymbol{\tau}'_{\text{min}} = -\boldsymbol{\tau}_{\text{max}} - \boldsymbol{\tau}_{\text{dyn}} - \boldsymbol{\tau}_{\text{ext}} \tag{3.6}$$

$$\boldsymbol{\tau}'_{\text{max}} = \boldsymbol{\tau}_{\text{max}} - \boldsymbol{\tau}_{\text{dyn}} - \boldsymbol{\tau}_{\text{ext}}. \tag{3.7}$$

Once a solution to (3.5) is found, the joint torque command to be sent to the robot is:

$$\boldsymbol{\tau}^{*} = \boldsymbol{\tau} + \boldsymbol{\tau}_{\text{dyn}}. \tag{3.8}$$

3.5 Deep Learning software

For our Learning from Demonstration experiments in our Chapter 6 we used Python 3 and Tensor Flow. The communication with the haptic feedback device was achieved using ROSpy, a ROS wrapper for Python. We had to develop from scratch the python ROS node for our Haptic Feedback device as the one provided by the manufacturer was not working with python.

CHAPTER 4

IMPEDANCE ESTIMATION AND TELE-IMPEDANCE

Contents

4.1 Introduction

Human physical assistance requires the assistant to tune both his trajectory and impedance in order to assist an individual as well as be guided by him. In this study we propose a controller for teleoperated human assistance that allows the assistant to guide the assisting robot in both trajectory and impedance. We propose to use the inherent perturbations in the task, for impedance estimation, while a simple neuroscience based filter allows the reference estimation of the operator. We tested our impedance estimation and the controller as a whole in two experiments in which a human operator guided a robot suffering force perturbations that simulated a human patient.

4.2 Background

In 2009, adults of age 65 or more represented 11% of the world population, and this percentage is expected to double by 2050 [UNR]. The percentage of elders above the age of 65 is 28% in the European Union [2016], and it is expected to reach 34% in Japan by 2030 [Muramatsu and Akiyama, 2011]. Elderly care and support, and specifically the lack of human assistants to help them, is a major concern for health-care, and in this

regard robots are seen as a promising tool [Broekens et al., 2009]. In this work, we are interested in robotic elderly physical assistance, in scenarios such as lifting the person out of the bath or chair, and for assistance in feeding, which have been identified as priority tasks in elderly care [2014].

A human physician or physical assistant can help a person stand up or take a cup to his/her mouth, in spite of arm tremor. In these interactions, the assistant is not (or at least, is not always) the 'leader' who imposes or forces the patient's movements. The assistant in fact acts as a 'collaborator', who aids haptically, while predicting and perceiving the motion intention [Ganesh et al., 2014, Kato et al., 2019, Takagi et al., 2018], and constraints of the other individual. Ideally, one would like a robot assistant to be able to do the same. However, this physical collaboration requires force and impedance adaptations, and prediction of the haptic behavior, all of which are non-trivial challenges for robots. And while researchers have proposed robot controllers which mimic human impedance adaptation [Li et al., 2018, Yang et al., 2011, Ganesh et al., 2010] and physical assistance [Takagi et al., 2017], these controllers are reactive, and need a predefined reference, that is difficult to anticipate in an assistive scenario. It will take some time before robots will be as effective as a human assistant.

Another way of replicating the human assistant's behavior on a robot is to include him/her 'in the loop', for example via tele-operation [Niemeyer et al., 2016]. In this case, the physical assistant drives the behavior of the assisting robot. This is the focus of our study. In regard to patient or elderly care, teleoperation cannot remove the requirement of the human assistant. Yet, it can aid one assistant help multiple individuals without going to every patient physically, hence it decreases the 'assistants over patients' ratio.

Teleoperation traditionally uses either an impedance or an admittance framework to connect the human 'leader' to the robot 'follower'; the impedance in these scenarios is either constant or adapted, but adapted relative to the environment, not the human operator [Niemeyer et al., 2016]. Instead, for human assistance, we need a control framework that allows the transfer of impedance as well as kinematic trajectories from the human operator to the assisting follower robot. We can try to achieve this with stiff position control, but the stability of such an arrangement is not possible due to limitations of the control frequency and presence of feedback and control delays that are typical of tele-operation setups [Mouri et al., 2017, Cortesao et al., 2006]. An alternate method one may think of is to estimate the desired/reference trajectory and impedance of the human operator and implement these as an impedance controller on the robot side. While this method still suffers from performance issues due to feedback and control delays, it can be passive and more efficient in terms of the stability. This however requires one to estimate the human impedance, as well as movement reference online during task performance.

The impedance applied by a human during a movement can be estimated either by perturbing the human limb [Gomi and Kawato, 1997] or by estimating muscle activation using electromyography (EMG) or grip force. Many recent studies have utilized EMG [Ganesh et al., 2010, Peternel et al., 2017, Luo et al., 2019, Ajoudani et al., 2012] or grip force[Walker et al., 2010], [Takagi et al., 2020] for human impedance estimation. Relying on muscle activation enables impedance estimation without the need for

external perturbations. Besides, the changes in EMG and grip force are not only due to the limb impedance (i.e., to the stiffness and damping parameters) but are person specific, and also due to: limb motion trajectory, body posture and time (fatigue). Therefore, while EMG or grip force may still be good methods to estimate impedance in the absence of perturbations, they require user-specific calibration [Ajoudani, 2016, Doornebosch et al., 2021]. On the other hand, in the presence of external perturbations, particularly continuous and non-repetitive ones, EMG and grip force signals include muscle reflexes, which are characterized by their own temporal and state dynamics [Doornebosch et al., 2021] making impedance estimation non trivial.

Impedance can be estimated by adding controlled perturbations [Hill and Niemeyer, 2009], but this can be detrimental for the task. Yet, in tasks like human assistance, which are themselves characterized by frequent perturbations, it is arguably better to utilize this technique – i.e., to estimate the impedance from the recorded perturbation forces and the resulting movement disturbances. Impedance measured from perturbations can be more representative – quantitatively and qualitatively – than the one estimated from muscle activation. Qualitatively, because it can enable better measures of directional impedance variations, and quantitatively because the measurement is directly at the human hand and it avoids the noise in EMG signals.

In this study, we propose a procedure for online (i.e., during the task) estimation of the human impedance from the perturbations. We will focus on the estimation of the stiffness and damping of the human operator, while assuming that the robot mass can be compensated for. We also propose a method, inspired by neuroscience, to estimate the reference trajectory of the human leader. Overall, our controller enables the transfer of force, trajectory and impedance, in the presence of unknown external perturbations. We test the controller in an embodied tele-assistance experiments.

The chapter is organized as follows. In Sect. 4.3, we present the tele-assistance framework, including human arm impedance parameters estimation and robot control. Next in Sect. 4.4, we will present three experiments. Experiment 1 is to validate our human impedance estimation procedure. Then in Experiment-2, we test the impedance estimation and controller in a maze task in which human operator was required to, in some scenarios, guide the robot through a channel (a task requiring high impedance) in the presence of external disturbances (which simulated a patient) in 1-dimension, and in other scenarios, follow the directions preferred by the robot (a task requiring low impedance). In this experiment we will neglect any feedback delays, and focus on the issue of impedance and trajectory transfer assuming that popular methods of delay compensation can be utilized as such. Finally in Experiment-3 we do a stress test of the system in a scenario with 2-dimensional perturbations as well as a visual feedback delay of 500 ms. We summarize and discuss the results in the 4.5 section before concluding in the section 4.6 .

4.3 Methods

Let us consider an operator using a haptic feedback enabled controller teleoperating a serial robot manipulator.

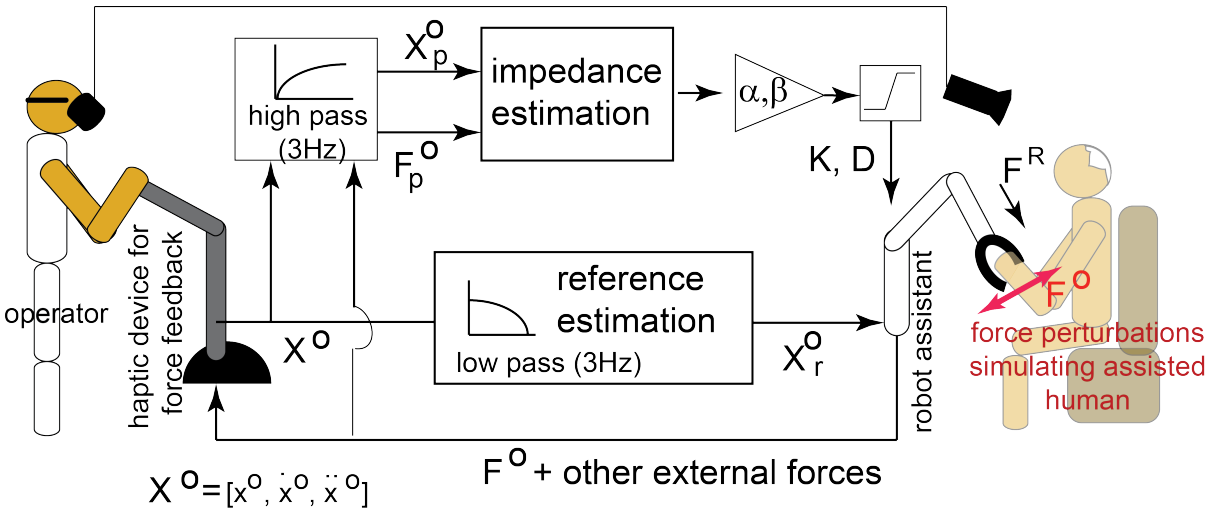


FIGURE 4.1 – *Block diagram of our proposed controller and setup. The key features are the robot reference estimation and the online perturbation based impedance estimation blocks.*

In such a scenario, keeping the operator's feeling of the system as transparent as possible is the main challenge, in order to properly transfer the operator's position, force and impedance. We model the operator's Cartesian arm impedance as a mechanical mass-spring-damper system with the following equation (4.1):

$$\boldsymbol{F}^{\boldsymbol{O}} = \boldsymbol{K}\Delta\boldsymbol{x}^{O} + \boldsymbol{D}\Delta\dot{\boldsymbol{x}}^{O} + \boldsymbol{M}\Delta\ddot{\boldsymbol{x}}^{O}, \tag{4.1}$$

where $\boldsymbol{F}^{\boldsymbol{O}} \in \mathbb{R}^6$ is the resulting force imposed on the human operator (through our haptic device) , $\boldsymbol{K}$, $\boldsymbol{D}$ and $\boldsymbol{M}$ are the positive-definite 6×6 matrices of stiffness, damping and mass, and $\Delta\boldsymbol{x}^{O} = \boldsymbol{x}_r^{O} - \boldsymbol{x}^{O} \in \mathbb{R}^6$ is the error between the reference and measured positions. Here the O superscript stands for Operator (i.e., the human leader).

The operator's spatial state $\left[\boldsymbol{x}_r^{O}\ \dot{\boldsymbol{x}}_r^{O}\ \ddot{\boldsymbol{x}}_r^{O}\right]$ is obtained using the encoder readings from the haptic device. The estimation of his/her arm's impedance parameters $\left[\boldsymbol{K}\ \boldsymbol{D}\ \boldsymbol{M}\right]$ is however not trivial. Several solutions have been proposed [Ajoudani, 2016, Hill and Niemeyer, 2009, Walker et al., 2010, Luo et al., 2019] but all require additional and potentially intrusive hardware. To cope with this issue, we propose a method to estimate in real time the operator's arm impedance parameters as well as the operator's 'desired states' or reference using only the data available from the haptic device. We rely on known properties of the human motor system to facilitate these estimations. These parameters can then be transferred to the robot's impedance controller, to better mimic the operator's behavior and increase the system transparency.

The whole procedure we use in our methods is described in (Fig. 4.1). We will start by describing the reference estimation procedure in 4.3.1, then how we extract the impedance parameters from the haptic device signals in 4.3.2, and finish with a description of the robot controller in 3.4.

4.3.1 Reference estimation

Human movements are enabled by the simultaneous modulation of trajectory, force and impedance [Wolpert et al., 2011, Ganesh and Burdet, 2013]. However, the modulation of each one has properties determined by the human body sensory and mechanical constraints[Etienne Burdet, 2013, Franklin et al., 2007]. Here, we utilize one of these properties with regards to perturbation regulation; it has been shown that humans compensate for lower frequency perturbations by using a synchronized and opposing 'reciprocal activation' i.e., a feedforward force. On the other hand, as the perturbation frequency increases, they increase 'co-contraction' – hence impedance – to compensate for perturbations, relying completely on impedance above a certain frequency threshold[G.Ganesh, 2020]. This is because while human generated forces (in the absence of impacts) can contain frequencies of over 10 Hz, the frequency of the controllable movements are much lower. While the threshold frequencies change depending on the limb in question, they decrease with the size of the limb. For the wrist, reciprocal activations fall to almost 20% of their values with a frequency of $3.5Hz$ [G.Ganesh, 2020]. Thus here we hypothesized $3Hz$ to be a suitable frequency threshold given that the perturbations disturb the whole arm in our setup.

The above observations provide us with two intuitions that help us with the reference and impedance estimation of the operator. First, because the reference trajectory is a component of the feedforward forces by the human operator, the lower frequency components of the operator states are more likely to represent his reference.

Indeed the need to differentiate the operator's reference (the target) and the operator's impedance changes is crucial. Let's take the case of a contact between the robot arm and the environment : in the case of this contact force suddenly increasing, it is not possible to deduce if the operator's reference moved toward the environment and hence the operator purposely augmented the contact force or if the operator just increased his own arm stiffness on the axis of the contact force, and as a result the reference moved away from the robot increasing the contact force

And second, for impedance estimation, we should consider the high frequency components of both the operator states and operator forces, because higher frequency components are more likely to be a result of impedance and not feedforward forces or reference changes.

We therefore split the observed variable on the operator side into two components:

$$\left[\boldsymbol{x}_r^O \; \dot{\boldsymbol{x}}_r^O \; \ddot{\boldsymbol{x}}_r^O\right] = \text{LPF}_3(\left[\boldsymbol{x}^O \; \dot{\boldsymbol{x}}^O \; \ddot{\boldsymbol{x}}^O\right]) \tag{4.2}$$

$$\left[\boldsymbol{x}^O{}_p \; \dot{\boldsymbol{x}}^O{}_p \; \ddot{\boldsymbol{x}}^O{}_p \; \boldsymbol{F}^{\boldsymbol{O}}{}_p\right] = \text{HPF}_3(\left[\boldsymbol{x}^O \; \dot{\boldsymbol{x}}^O \; \ddot{\boldsymbol{x}}^O \; \boldsymbol{F}^{\boldsymbol{O}}\right]). \tag{4.3}$$

In these equations, the r subscript denotes the reference whereas the perturbed component of the spatial state, denoted by subscript p, will be used for impedance estimation, as we explain in the next section. LPF_3 and HPF_3 are respectively low and high pass filters with cutoff frequency 3 Hz.

4.3.2 Impedance estimation procedure

The objective of the impedance estimation is to derive $\boldsymbol{K}$ and $\boldsymbol{D}$ from (4.1), knowning force $\boldsymbol{F^O}$ generated by the haptic device, spatial state $\begin{bmatrix}\boldsymbol{x}^O & \dot{\boldsymbol{x}}^O & \ddot{\boldsymbol{x}}^O\end{bmatrix}$, reference state $\begin{bmatrix}\boldsymbol{x}_r^O & \dot{\boldsymbol{x}}_r^O & \ddot{\boldsymbol{x}}_r^O\end{bmatrix}$ given by (4.2) and a priori effective cartesian mass $\boldsymbol{M}$. We consider the mass of the operator to be constant, under the assumption that his/her body and arm posture do not change significantly during operation. $\boldsymbol{M}$ was taken to be equal to $1Kg$ in line with human arm reach modelling studies [Wolpert et al., 2011]

Then from (4.1), we estimate $\boldsymbol{K}$ and $\boldsymbol{D}$ using least squares fit over a window of n consecutive samples:

$$\boldsymbol{A} = (\boldsymbol{F} - \boldsymbol{M}\ddot{\boldsymbol{X}})\boldsymbol{J}_O^{\dagger}, \tag{4.4}$$

with:

$$\boldsymbol{A} = \begin{bmatrix}\boldsymbol{K} & \boldsymbol{D}\end{bmatrix} \tag{4.5}$$

$$\boldsymbol{F} = \begin{bmatrix}\boldsymbol{F^O}_p \dots \boldsymbol{F^O}_{p_n}\end{bmatrix} \tag{4.6}$$

$$\ddot{\boldsymbol{X}} = \begin{bmatrix}\ddot{\boldsymbol{x}^O}_p \dots \ddot{\boldsymbol{x}^O}_{p_n}\end{bmatrix} \tag{4.7}$$

$$\boldsymbol{J}_O = \begin{bmatrix}\Delta\boldsymbol{x}^O{}_p \dots \Delta\boldsymbol{x}^O{}_{p_n} \\ \Delta\dot{\boldsymbol{x}^O}_p \dots \Delta\dot{\boldsymbol{x}^O}_{p_n}\end{bmatrix}, \tag{4.8}$$

and $\boldsymbol{J}_O^{\dagger}$ denotes the Moore-Penrose pseudo-inverse of $\boldsymbol{J}_O$.

4.4 Experiments and results

4.4.1 Experimental Setup

We used three experiments to verify our stiffness estimation procedure and the tele operated assistance system. The setup is depicted in (Fig. 6.5). It consisted of a 7DOF robot arm Franka Panda and a haptic feedback device Haption Virtuose 3D which can feedback 3 linear forces. We utilized a HTC Vive Pro HMD with a 360 degree camera to make the operator see the task from the same point of view as if the robot was his/her own arms, i.e. as if the robot was embodied [Toet et al., 2020a]. To verify the correctness of the impedance estimator, we recorded Electromyography (EMG) in Experiment-1, with the Delsys Trigno wireless EMG.

During the experiments, the stiffness estimated from the perturbations was smoothed by a Butterworth low pass filter at 0.5 Hz (second order). On the other hand, we could not use the damping parameters calculated on the human operator and had to use the critical damping value calculated relative to the stiffness as $\boldsymbol{D} = 2\sqrt{\boldsymbol{KM}}$ (with Mass =1 Kg). We found that, probably due to the lack of an accurate mass compensation on our robot, the human calculated damping parameters were not sufficient to ensure stable performance. For security reasons, we also limited the robot stiffness values between 100 N/m and 10000 N/m for each Cartesian axis.

4.4.2 Experiment 1: Verification of our stiffness estimation

We started by verifying the correctness and resolution of our stiffness estimation. Unfortunately, for the ground truth, we had to rely on muscle activation, hence electromyography (EMG), which as mentioned before suffers from various limitations related to movements. To overcome them, we asked the participant to maintain a static arm posture while being disturbed by force perturbations from the haptic device. Furthermore, we chose to make the perturbations repetitive (albeit at a high frequency of 3.5 Hz, so that the participant could not compensate via feedforward forces). The measures ensured that a constant muscle activation (hence a specific EMG level) would represent a constant impedance at the hand. In this scenario, the participants were provided with a feedback of the estimated stiffness on a computer screen (see inset of (Fig. 4.2)) while we asked them to maintain their stiffness at different target levels. We recorded EMG from four muscles in the arm (Biceps Brachii, Triceps Brachii Lateral Head, Flexi Carpi Radialis and Extensor Carpi Radialis) that were expected to contribute to the task space stiffness of the hand in our task (see plots in (Fig. 4.2)). While the EMG levels do not directly give the absolute impedance of the hand, the total EMG level (represented by the smooth envelope in (Fig. 4.2)) is known to correlate with the impedance, and stiffness (assuming the damping correlates with the stiffness at the hand) of the hand. We could observe different muscle pairs activating when the participants controlled their hand stiffness in the X (left column of (Fig. 4.2)), Y (middle column) and Z (right column), while the total EMG was found to co-vary with our estimated stiffness in each case.

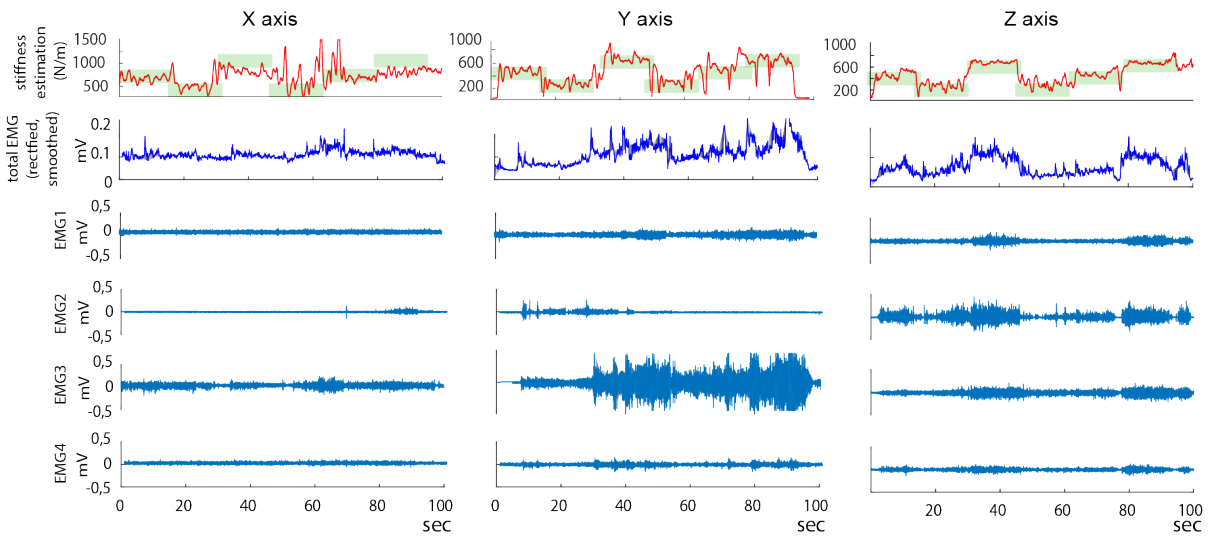


FIGURE 4.2 – *Experiment 1, validation of our stiffness estimation. The participant held the Virtuose haptic device in the presence of perturbations and was provided with a feedback of the estimated stiffness by our algorithm. He was required to match his stiffness to target values displayed on the screen, represented here by cyan rectangle area in the estimation plot. We compared his stiffness changes in X (left), Y (middle), and Z (right), with the EMG recorded on four muscles and the total rectified EMG, representative of the arm impedance level. Note that the EMG just served the purpose of validating the correctness of the stiffness changes estimated by our algorithm an will not be used in our system for robot control.*

4.4.3 Experiment 2: Controller verification during tele-assistance

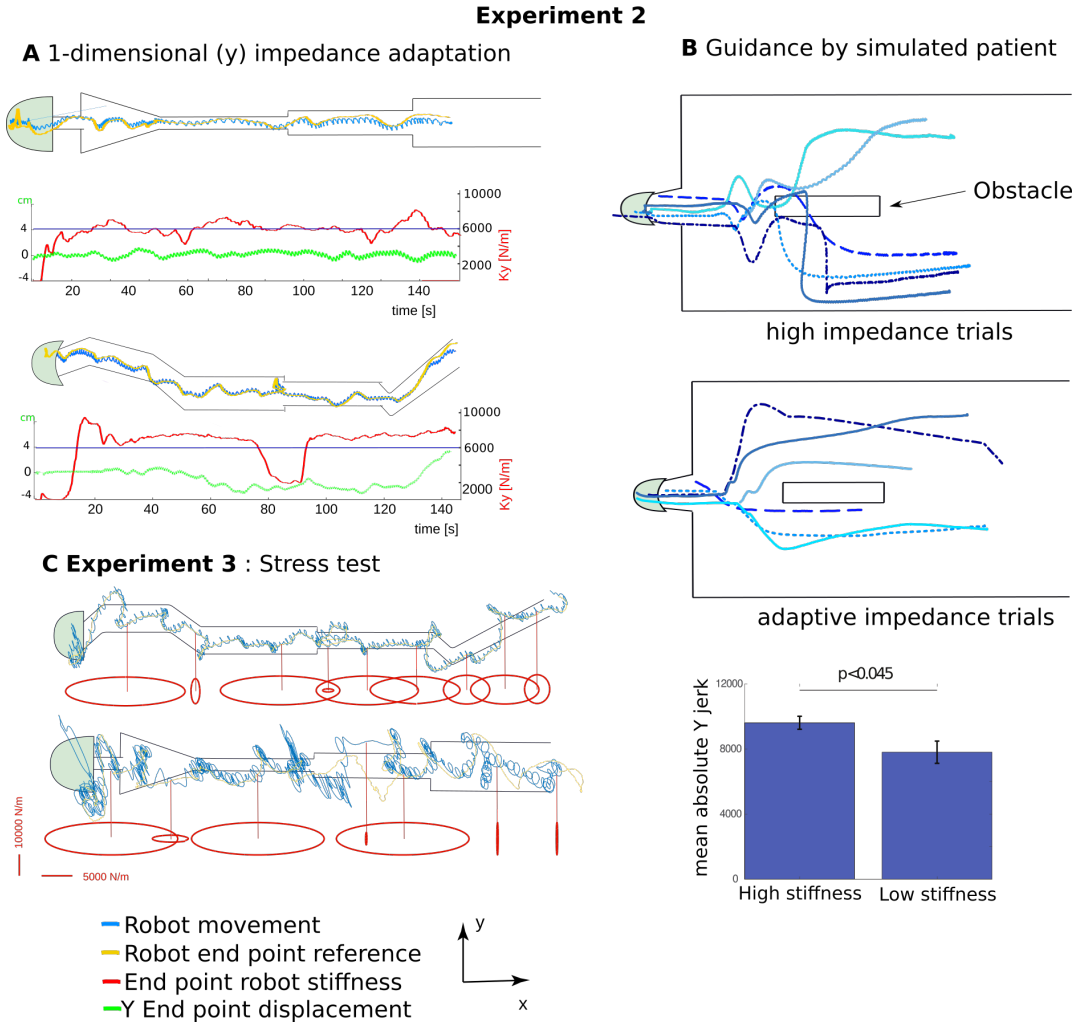


FIGURE 4.3 – *Experiment 2, Assistance task. In the assistance task the operator was asked to guide the robot with a pen through a maze (starting with the light green semi-circle and defined by the black walls) while remaining inside the walls. Force perturbations on the robot simulated the disturbances from an assisted elderly individual. We performed the two experiments. Experiment 2A) The perturbations were in one dimension (Y) and the operator had to guide the robot while regulating his y- impedance to keep the robot within the walls. 2B) The operator was blind folded and asked to assist a simulated patient (simulated by the forces on our robot) while being haptically guided by the patient to avoid an obstacle. The operator worked in trials where the robot impedance was prefixed at 6000 N/m (upper panel) or estimated from the operator (lower panel). The operator did not have prior knowledge of the condition or the direction of guidance. The average magnitude of y-jerk, observed between 0.5 seconds and 2 seconds after the guidance force was initiated, was significantly higher in case of the fixed impedance trials ($p < 0.045$, 2 sample T-test). Error bars represent standard error. C) Experiment-3 served as a stress test for our system in which we introduced perturbations in two dimensions and there was a 500 ms delay in the visual feedback provided to the operator. The stiffness ellipses calculated in the x-y space are shown in red and connected (with a thin red line) to the position in space where they were calculated.*

Next, in Experiment-2, we verified how our controller performed in an assistance task and how the behavior differed when the robot impedance was kept constant. Ex-

periment 2 had three conditions. In each condition, a 'operator' assistant guided a robot, while it helped a patient draw a line with a pen through a maze (starting with the light green semi-circles and defined by the black walls, fig 4.3). We did not have a real patient in the task. The patient's perturbations were simulated by force perturbations imposed on the robot, and the pen was held by the robot's two-fingers gripper. The type of perturbations and hence the impedance adaptations required by the operator were varied across the three conditions (fig 4.3 A, B and C). Parameter α was set to 30 in Eqn. (3.4).

A) One-dimensional (Y) perturbations ((Fig. 4.3)A): We started with patient perturbations only along the Y dimension, perpendicular to the required pen direction. The perturbations were sinusoidal, with a Frequency of 3.5 Hz and 10N amplitude. The operator was able to control his impedance (red traces) to regulate the movement of the robot through the maze (blue traces). The Y displacement in time is also plotted (green trace).

B) Adaptive vs Fixed impedance ((Fig. 4.3)B): Different human assistance task require different impedances. A task requiring the human-operator to both guide (in direction) and assist (against perturbation) a patient requires higher impedances (like in our above experiment), but when the guidance is expected from the patient, better assistance is possible when the impedance of the robot is low. This variation is not possible if we use a fixed impedance on the robot. To show this, in Experiment-1B we created a scenario where the human operator assists according to guidance from the patient (again simulated by forces on the robot). Experiment 1B required the human operator to close his eyes and guide the robot, while a second experimenter applied a programmed push on the robot (9 Newton force pulse in y-direction applied for 200 ms) in either direction to guides the human operator away from an obstacle he would otherwise collide with. This scenario was repeated 12 times for the 2 directions X 2 impedance settings (K=6000 N/m) of adaptive (estimated from the operator)X 3 repetitions (see (Fig. 4.3)B). The human operator was unaware of which impedance setting and which direction of perturbation came in each trial. We calculated the absolute mean jerk in the y-direction in the trials and observed that the mean jerk in the 2 seconds after force perturbation was significantly higher for the fixed impedance condition) see bar graph in (Fig. 4.3)B).

4.4.4 Experiment 3: Controller stress test

Finally in Experiment-3 we made a stress test of the impedance estimation and the human- in loop controller. We introduced two changes in the task of Experiment-2A. First, the operator was subjected to 2-dimensional random perturbations and he had to adjust his impedance in two axes during the task. Second, the operator visual feedback was subjected to a delay of 500 ms.

The results are shown in (Fig. 4.3)C. Though the operator found the task quite difficult, especially because of the visual delay, crucially we could verify that we could measure and modulate the robot impedance in two dimensions. The X-Y stiffness values during the task are plotted as stiffness ellipses which represent the estimated force

for unit displacement in every direction. Note that the Eigen direction of the stiffness ellipses remain the same (while they change only in magnitude) because in this study we assume the task space $\boldsymbol{K}_x$ and $\boldsymbol{K}_y$ to be independent, and we do not consider the off diagonal terms in Eqn. (4.5).

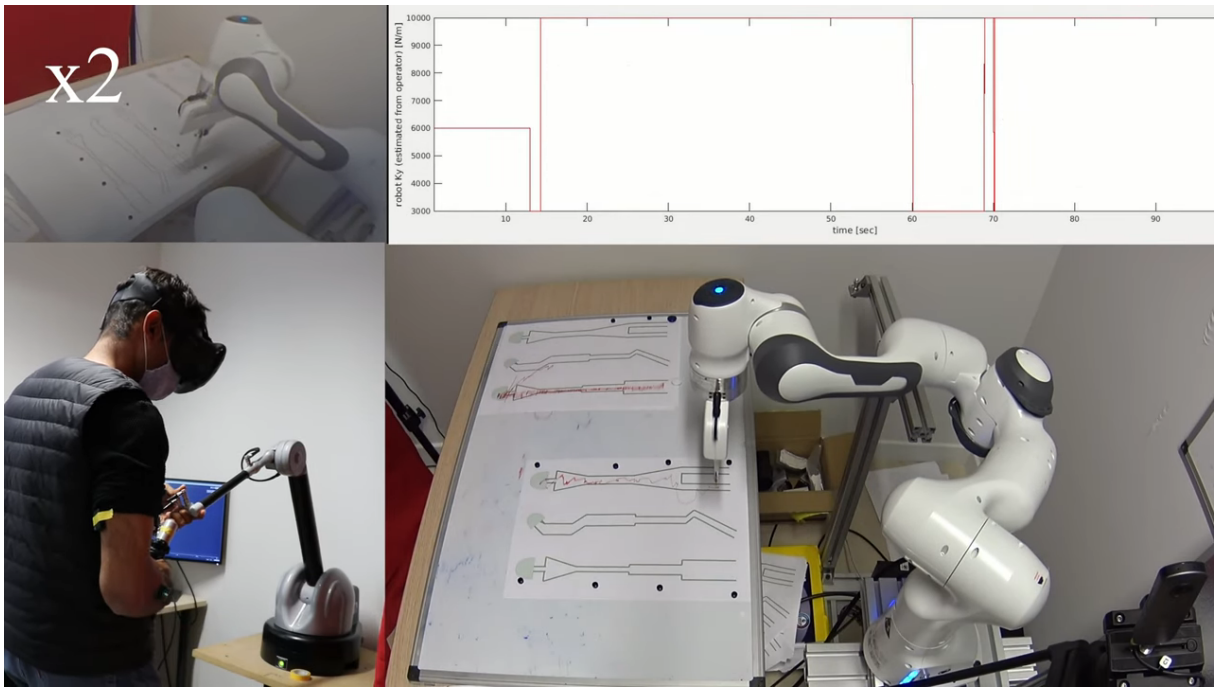


FIGURE 4.4 – *Youtube Video of our impedance estimation experiment (the QR code is a clickable link)*

4.5 Contributions

In this study, we presented a human guided impedance controller during teleoperation. We designed the controller envisaging use in assistive scenarios, where a human physiotherapist or caregiver guides a follower robot to help elderly patients. These scenarios are characterized by perturbations from the individual and hence we propose to estimate the human impedance directly from the perturbations. For this purpose, we propose a methodology for online impedance estimation, and online reference estimation from the human operator.

As mentioned in the experiments, we were unable to use the damping calculated from the human operator on our robot. We found these values too low, relative to the stiffness, to ensure stable robot behavior. This comes as no surprise, given the different inertias of human arm and robot. Ideally our controller, in which the robot forces are fed to the operator and the operator movements are sent to the robot, should

impose the human arm dynamics on the robot, therefore theoretically ensuring that the human damping ratios are sufficient for the robot. In practice though, this is possible only if the mass of the robot is well compensated for, which is not a trivial challenge. From human studies we know that human damping increases monotonically with stiffness[Etienne Burdet, 2013, Franklin et al., 2007]. Given these observations, tuning the damping separately, like we did in our experiment, seems to be a quick and sufficient solution for assistive tasks. However, further studies are required to clarify this issue.

Human interactive behaviors are enabled by simultaneous adaptations of force, trajectory and impedance. These adaptation are both predictive, to ensure stability when an interaction starts, as well as reactive, to maintain the stability during perturbations in a task. The method we propose here is specific for the measurement of reactive impedance and is arguably better than muscle activation (EMG) based impedance estimations in the presence of perturbations (as discussed in the introduction). On the other hand, muscle activation based techniques are the only ones available for impedance estimations before the start of a movement, and in the absence of sufficient external perturbations. Robust impedance estimation in real world tasks therefore requires us to develop an integrated estimation framework in which the predictive impedance changes can be measured using muscle activation (via EMG or grip force) and the reactive changes are estimated using the perturbations, like we propose here in this study.

4.6 Conclusion

In conclusion, here we presented a methodology for impedance control during teleoperation with estimation and transfer of the reference and impedance from the human operator, to the robot. We provide a method of online human impedance estimation using the perturbations inherent in the task. This first work provided the first step towards an assistive teleoperated system for possible human assistance in the future.

CHAPTER 5

DEEP LEARNING BACKGROUND

Contents

5.1 Introduction

In our previous work we set-up an embodied robot system, this chapter will be dedicated to the Machine Learning and Deep Learning background needed to understand the next chapter that will bring Learning from Demonstration in our embodied robot system. We focused our learning to be computer vision based and labeled with human expert produced motions.

This chapter will be dedicated to explain Learning concepts, architectures and Learning based Computer Vision related knowledge.

5.2 Differences in Machine Learning methods

Given the previously cited objectives it would be possible to think about a lot of different possible uses of Machine Learning, for example, policy learning and one shot learning, Reinforcement Learning (RL), Inverse Reinforcement Learning(IRL), Artificial Neural Networks (ANN), MultiLayer Perceptron (MLP), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) like Long Short Term Memory (LSTM), networks and Gated Recurrent Unit (GRU) or even state of the art Transformers.

5.3 Choices of architectures

One of the main choices we have made is to learn only from data that the human operator directly senses or produces. We did not want to model our task but we wanted the human to decide by themself given a task how to solve the problem as they would want to. By doing this it would be possible to learn different aspects of the human way to solve the task. But this decision restrained us on the type of network we could use. For example, using Reinforcement Learning needs a simulator to generate data would not be possible anymore, since our data should come only from the human. This led us to use principally Neural Networks and vision based networks as the main input would be pictures.

In the next sections we will detail the basic architectures used in Deep Learning. Some of these architectures are not directly used in our work but the principles they are based on are used in our work, so we estimated it would be needed to clarify some points.

5.4 Terminology

We will first explain the most commons terms needed for understanding machine learning, more specifically terms used in our work, needed to understand data structures, learning different phases and to clarify some common points that might be confusing.

- **Shape** : the shape of any data, is the size of every array needed to represent it.
  The shape of a pose array is (6,)
  The shape of a HD colored picture is (1280,720,3)
  The shape of a HD black and white video is (1280,720,1,W) with W the number of frames.

- **Dataset** : when collecting data, the dataset is the whole data that will be used for the training. It contains on one side the data the user wants to train on, which

can be numerical signals, images, time-series, in 1 dimension or multiples. And on the other side, if any, the labels that the user wants to compare the prediction with, it can be the exact same type of data as cited before or classes, in classification cases. Every set of data in the data-set must correspond to one set of label The total data-set size is called DS.

- **Input**: the Network's input is defined as the data, of one or multiple types, one wants the network to train on.

- **Output**: the Network's output is, given an Input, the result of the network. The size and type of the output will be defined by the label's shape.

- **Label**: the label refers to some data in the data-set, that will be used as a ground truth used to compute the loss function. Ideally in a perfect network the output should be equal to the the label.

- **Supervised and unsupervised learning** : In Machine Learning we can differentiate two different type of learning, one where the whole dataset is labeled, in this case it's called supervised learning, and for unsupervised learning where the dataset is not labeled and the aim of the learning is to find pattern in the non labeled dataset.

- **Epoch** : during a training, an epoch corresponds to the fact that the learning algorithm has been input exactly once with every data(or series of data) present in the data-set.

- **Batch** : during one epoch, the gradient descent is not done on the whole data-set, it would mean to load the whole data-set in the RAM or the VRAM which would be impossible in some cases. Instead it is generally welcome to cut the data-set in N different little data-set called batches. The gradient descent will be then applied to every batch in order to complete one epoch. The batch size BS is defined as $N = DS/BS$. In the case there is a rest in the division, the leftover data can be used in a non standard-sized batch or left out by the user.

- **Nested Batches** : when working with time-series, it is generally needed to not only use as an input of the network a specific data but a window WS of data around a specific time value. It is then needed to create batches of this window in the data-set and to apply an increment i to this WS on the dataset in order to get the nested batches. In the case i is inferior than WS some data will appear multiple times in different Nested Batches. In the case Nested Batches are used, one batch is not anymore composed of data taken from the data-set directly but composed of BS Nested Batches.

- **Step** : Every batch is composed of BS steps, every step takes the corresponding input data from the data-set, the corresponding label and compute the loss function to output some loss.

- **Training** : In learning the phase when a loss function, the gradient descents are applied to the data-set in order to update the weights of the network is called the training.

- **Prediction** : Once the model is trained, the prediction phase consists in applying the model and its weights to the new input, that were never seen by the network and to obtain an output that represents what the network has learnt

- **Validation dataset** : During the training phase, at the end of every epoch, a prediction the validation data is done. This validation data is a small but effective percentage of the data-set (generally 20 percent) that was taken out from the data-set before the first epoch started to be trained. By doing this the user can after every epoch check the prediction accuracy of the current training on new data, never input during the training phase.

- **Normalization** : normalize the data will bring all the different variables in the same numerical range. This step is generally very important so that every input has the same range as the network weights.

- **Regression and Classification** : in supervised learning, there are two ways to label data. The first one is to label each data with other data, be it an image, a tensor, a float; this is called regression. In which case the output of the network will have the dimension of the label.
  The second one consists in labeling with classes, which corresponds to a predefined type. It is generally the case when a whole set of action is predefined or when one wants to differentiate cars and buses for example. The output will be a probability of confidence in each class used during training

- **Loss Function** : the loss function is a function that computes error between the label y_i and the output $\hat{y_i}$ for every data in the dataset. One of the loss function generally used in learning is the Mean Square Error :

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y_i})^2 \tag{5.1}$$

  with $\hat{y_i}$ defined as the estimation of the labeled ground truth y_i for the ith data and within N, the number of batch size in one complete batch.

5.5 Fully Connected layer

A Fully Connected (FC) layer also called Dense layer is a Neural Network Layer composed of n weights all connected to all the weights of the next layer, as the name implies.
The size of a fully connected layer output is then the number of weights n.
Each weight has its own value that the gradient descent will modify according to the result of the loss function.
Once the weights have been properly trained the Output of the Fully Connected layer should match the expected values for at least some given Input (Fig. 5.1).

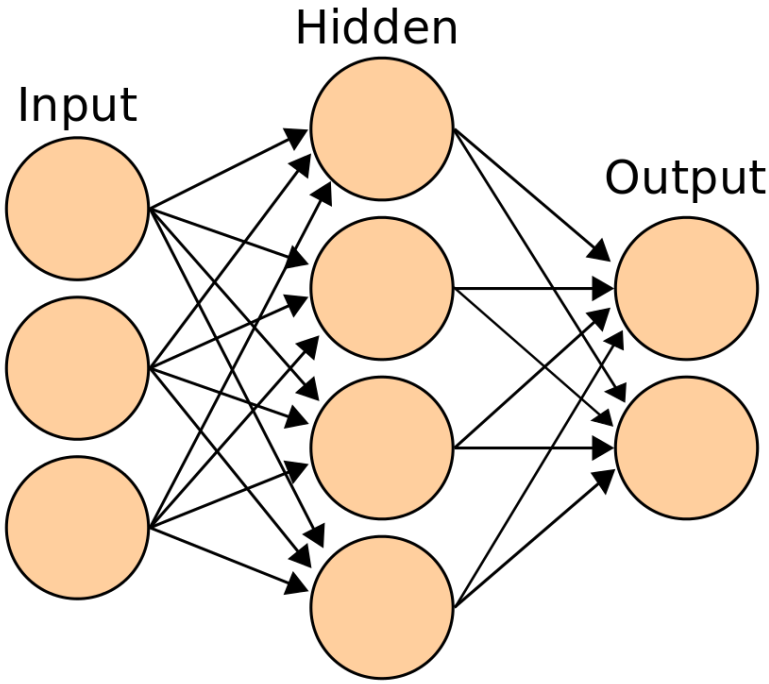


FIGURE 5.1 – *Example of a 2 Fully Connected layers network : the first one called "Hidden" gets 3 inputs from the previous "Input" layer and the "Output" Layer gets 4 input, the output from the "Hidden" layer*

5.6 Convolution principle and convolutional layer (CNN)

Convolutions are the basis of computer vision and visual recognition.
In order to explain quickly what applying a convolution filter K to an image I means we will use the figure (Fig. 5.2)
The convolution is applying a kernel K (also called filter) step by step by a pre-determined step size (called stride), all over the image by starting by the up-right corner. The kernel is sliding column after columns then changing row until it has reached column repeating this process up til the bottom-left corner
For every step, every pixel on the image is multiplied element by element by the corresponding kernel value and are then replaced in a new matrix to obtain the output for

the current location.

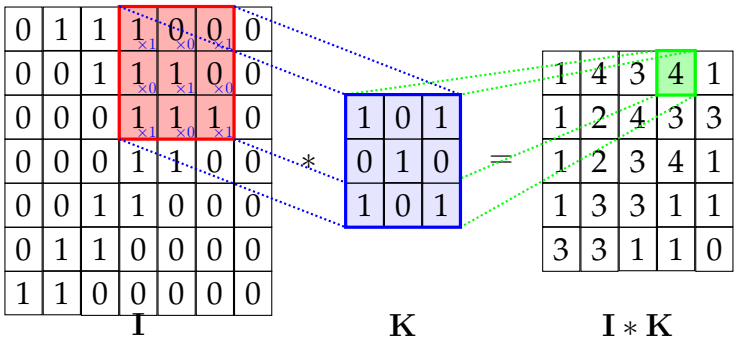


FIGURE 5.2 – *A convolution kernel **K** is applied to the input image **I**. Here the kernel size is 3x3*

The idea to apply convolutions to Neural Networks was first introduced in the 1980s by Yann LeCun. This work was based on the work from Kunihiko Fukushima in 1979, the neocognitron[Kunihiko, 1979], the first basic neural network using cells to extract features from the input.
A CNN layer applies n convolutions to the input image or matrix. It uses hyper-parameters like the padding size and the kernel size to apply every convolution as would do a normal convolution filter at the difference that the user will define a number n of filters to apply.
This will output n filtered images that will be fed to a fully connected layer. Then weights will be computed for each of the n filtered images to the output to find which kernel filters work best on the input image. The objective is to find which convolutions parameters works best to find features in images.

5.7 Pre-trained CNN : The example of VGG

A CNN can be trained on a lot of types of pictures to extract features but the idea to train CNN on millions of different images to get a "general" image recognition network has been done multiple times. We will explain the case of VGG, Very Deep Convolutional Networks for Large-Scale Image Recognition [Simonyan and Zisserman, 2015]. VGG was trained over 1.3M images and over 1000 classes of objects. It consists in stacking convolution layers, pooling layers and fully connected layers while decreasing the width of the feature and increasing its depth as shown in (Fig. 5.3)
(Fig. 5.4) gives an idea of a comparison of performances between different pre-trained networks.

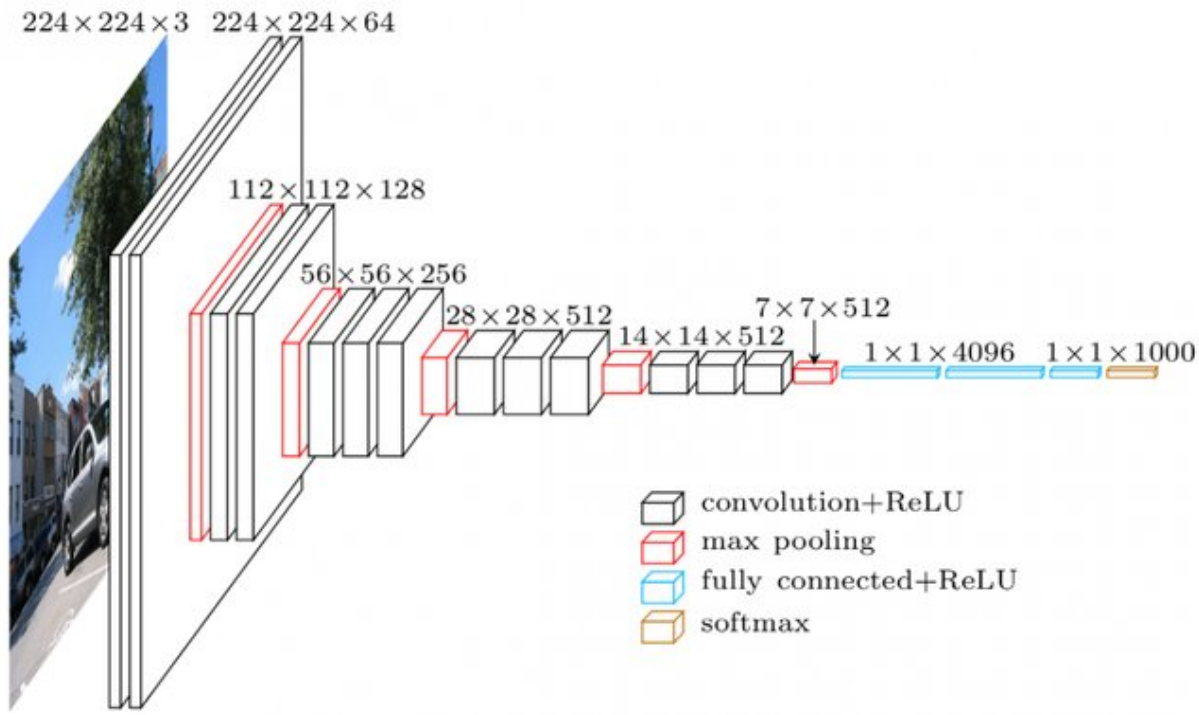


FIGURE 5.3 – *Complete VGG-16 Architecture Source : [Loukadakis et al., 2018]*

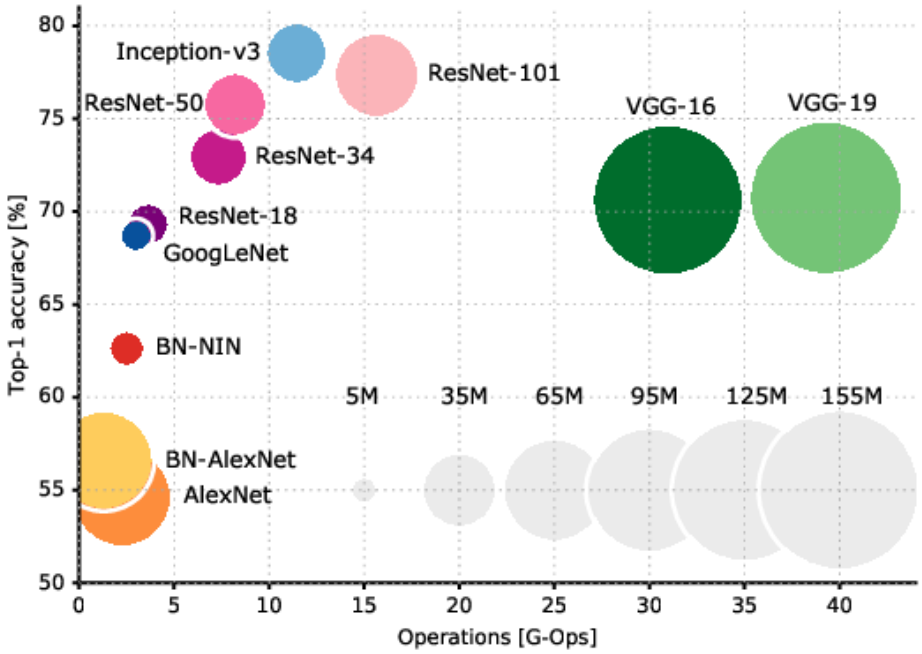


FIGURE 5.4 – *Different pre-trained image recognition networks, by comparing their accuracy with their number of weights. Source : [Canziani et al., 2016]*

5.8 Recurrent Neural Networks (RNN) and Long Short Term Memory (LSTM) units

Recurrent Neural Network (RNN) is a kind of network that is able to learn temporality or trajectories over time or space. With these aspect it is a good approach to deal with human motion [Hug et al., 2018]. These networks have shown success in Language Processing for generating text (Sutskever et al., 2011), hand written characters (Graves, 2013; Gregor et al., 2015), and even captioning images (Vinyals et al., 2014).

The Long Short Term Memory (LSTM) is a kind a RNN first developed in 1997 by Sepp Hochreiter and Jürgen Schmidhuber [Hochreiter and Schmidhuber, 1997] in order to avoid vanishing gradient problems and make possible both long term memory and short term memory in the same architecture. Then Gated Recurrent Unit (GRU) came as an evolution from LSTM in 2014 by Kyunghyun Cho [Cho et al., 2014] Contrary to the RNN the LSTM/GRU does not take as input only the previous state and the current input but it takes also N memory cells states as an input. N is the number of cell units defined in the architecture replacing the activation function of the RNN. The LSTM are being widely used in the computer vision and robotics fields, they were used by OpenAI to achieve dexterous movement to solve Rubik's Cube with the robot Shadow hand [Ope]

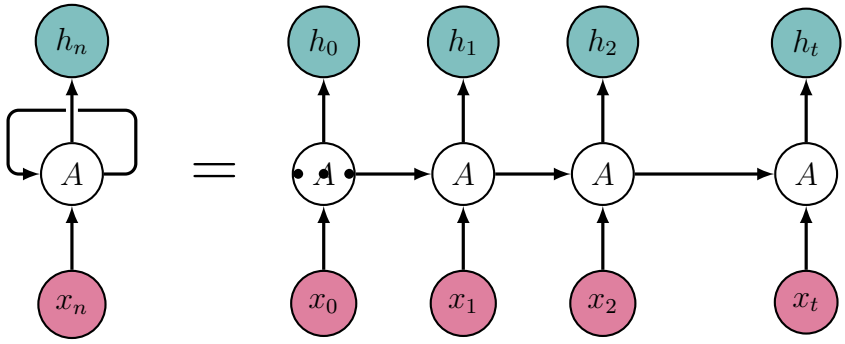


FIGURE 5.5 – *Composition of a RNN Architecture, x_n represents the input, h_n represents the output and A the internal Architecture*

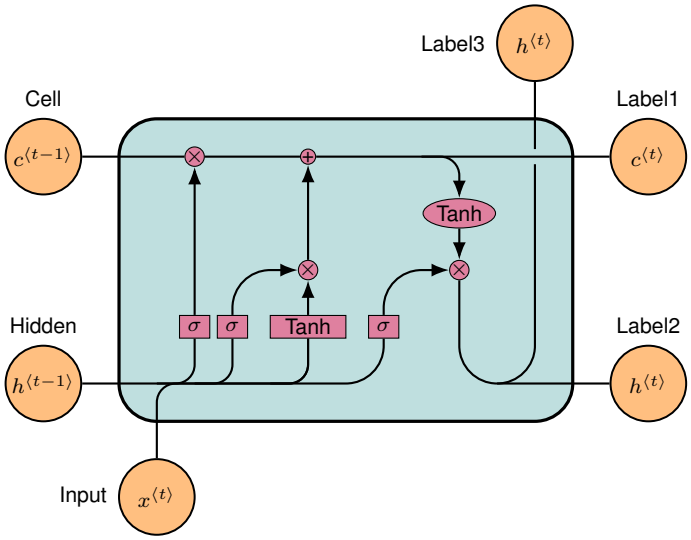


FIGURE 5.6 – *A type of RNN Architecture : LSTM internal architecture*
x_n represents the input, c_n represents the "memory", the output coming from the precedent unit/cell, h_n represents the output

5.9 YOLO

YOLO means You Only Look Once [Redmon et al., 2016], and is a Deep Learning backed Computer Vision algorithm. As the names implies, it takes the whole image as an input instead of multiples little images of the input image which makes it quite faster than other learning based algorithms. It is mainly used for object recognition like faces, cars etc in images and tracking in videos from classes on which it was trained before. Its strength is that it does not only output a confidence over classes but also a position and an area where this class might be present in the image.

5.10 Choosing the labels

In Deep Learning labeling will define the output of the network and in some case the architecture of the network. In our case, the Learning from Demonstration gives constraints related to robotics constraints, like the fact that data needs to come from a sensor. It could be a picture from a camera, a depth-map, a 2D or 3D distance map from a LIDAR, a time serie data from an All-or-Nothing sensor, a force/torque sensor, a recording of motion in the Cartesian space or the joint space from a robot, etc. These dataset are generally represented in their own frame, which might bring different problem.

Once the desired label chosen, it is first preferable to check the dataset for this label. In fact if this dataset is biased it will not lead to proper learning, similar to when learning from only a few examples of one class would lead to biases in classification. It can be seen in a simple example : let's say I want to learn from a force sensor that measures impacts, my dataset will be mostly composed of constant data and only in some places the force value will change to represent the impact. In this case the dataset will be biased because a simple regression will tend to learn the omnipresent constant value, and will have a really good accuracy meanwhile it will not have caught what it was trained on.
Some ways to work with biased datasets exist, like overweighting only the less represented samples, but they are hard to tune so we avoided using them.

Also, in the case the label has a particular shape, like pictures or videos, it is possible that the architecture of the network in itself needs to be adapted. Indeed in case of next frame prediction, in other words in order to synthesize new data, adding a classic Fully Connected layer in the network would "break" the shape of the data and it is likely the result of the training would not be as desired. In this case it might be better to add layers of time seried convolution or Convolutional LSTM in case of pictures.

It is of course possible to label with multiple label that would represent a position

in joint or cartesian space but it also possible to label with a time serie of these position in order to learn some path planning for example or some multiple next step ahead prediction.

Lastly, it is also better to check if the data does not contains any Not a Number (NaN) values in the dataset. It is unlikely but possible due to an overflow of some system when the data was recorded, some division by 0 or simply an ASCII character or a string (possibly like a space character) present in the dataset. This would get the gradient descent to propagate NaNs in the weights.

5.11 Dropout rate

The dropout layer is a layer that is applied after most types of layers, such as fully connected layers, convolutional layers, and recurrent layers. It acts as a reset on the output of the layer its applied to, therefore given a probability of occurrence chosen beforehand (the dropout rate) the dropout layer will attribute 0 to the whole output of specific layer its applied to. As explained in [Srivastava et al., 2014] it is useful to avoid over-fitting during the training phase because the dropout will constrain the network no to be too much confident in one particular example, which is especially the case when network are really deep compared to the number of examples in the dataset.

It is important to note that during prediction, the dropout layer are not present anymore in the model, so they do not interfere with the result, they are just effective during training.

5.12 Learning Rate (LR)

The Learning Rate (LR) is the rate at which it's possible to increase or decrease every weight of the network during the training phase. This rate is comprised between 0 and 1, with generally initial value comprised between 0.01 and 0.0001.
In order to understand what influence the learning rate has on the gradient descent it is generally explained in term of optimisation because the learning rate is also the "step size" of each step during the gradient descent.
The gradient descent will start from the user defined value, in the case it was set too high, it might step over the minimum. If it was set too low it will be stuck in any local minimum and will never converge to the theoretical global minimum or a more favorable local optima.

When choosing the value of the initial learning rate, it seems there is no perfect value to start with, it is empirical, like finding the right gain for a PID. It is also known that the more the network has trained on an important number of epochs the smaller should be the learning rate. We then need some function to decrease the LR after every epoch end, every epoch up until the end of the training.

There are few ways to change the LR over time, as choosing an exponential decrease over the number of epochs, or a decay scheduled after a fixed number of epochs but in our work we chose to use the Cosine Decay method [Loshchilov and Hutter, 2016], a decay method that proved its efficiency on multiple dataset like CIFAR, EMG times series dataset or a downsized version of the ImageNet dataset (see (Fig. 5.7)).

This method consists in having a steep change in LR every epoch following a cosine-like curve. By doing this, it allows the network to apply bigger changes to the weight periodically regarding the number of epochs that have already passed. It allows the gradient descent to once again go from another starting point in the case it was stuck in some local minimum for example.

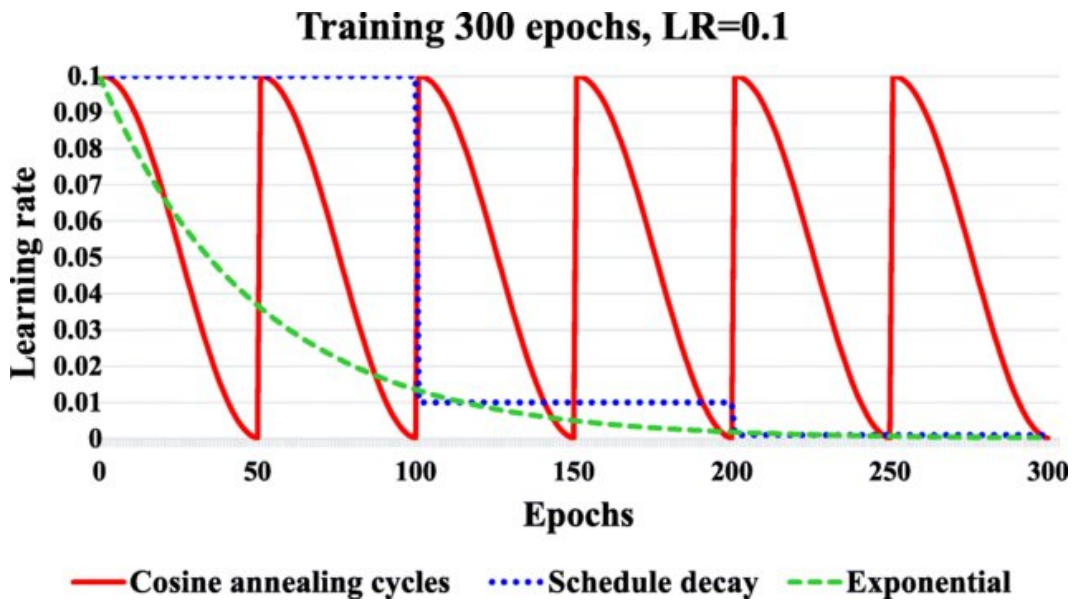


FIGURE 5.7 – *Comparison of 3 Learning Rate (LR) decay methods : the schedule decay, the exponential decay or the cosine decay. With this shape the Cosine method can avoid converging too much in a not optimized minimum. Source : [Termritthikun et al., 2019]*

CHAPTER 6

DEEP LEARNING BASED SKILL TRANSFER

Contents

6.1 Introduction

Up to this chapter we have developed an embodied robot system and it's control. As we conjectured that human behavioral data could lead to novel learning from demonstration approaches, hence, we used our embodied system to achieve skill transfer from human to robot. Indeed, the popularity of machine learning, like reinforcement learning, learning from demonstration, during this decade increased incommensurately due to the decreasing costs of GPUs and the availability of "big data". In this Chapter we use Learning from Demonstration in a teleoperated embodied system in order to teach an agent some human behavior.

6.2 Methods

In this work we aimed to learn from demonstration in the embodied robotic teleoperation system described before. We believe that using embodiment during a teleoperation task and learning from the data generated can bring more to *learning from demonstration* (LfD) due to several advantages it provides- First, in an embodied teleoperation system where the human is provided with first person perspective, and sufficient feedback, allows us to utilize the human cognitive skills to the full extent to solve the constraints presented by the robot (kinematics, joint limits, dynamic constraints as well as feedback constraints). Second, in an embodied setting all the feedback available to the human passes through our embodied robot system, and hence observable. Similarly, every action made by the human to control the robot also passes through our embodied robot system. An embodied teleoperation setup is hence ideal for using machine learning algorithms to map the sensory –motor processes defining the human cognitive skill. Finally, embodied teleoperation has been shown to improve human performance [Toet et al., 2020b, Iwasaki et al., 2022, Ventre-Dominey et al., 2019], pushing the case that embodiment may help human behaviors while teaching robots.

To demonstrate this idea, here we chose a maze task and used a visual recognition-based deep learning Agent to try to learn 'human like' movements in the maze. We modified the dynamics presented to the humans by adding two velocity dependent (damping) force fields, which lead to changes in their behavior in these mazes. We show that in each case the Agent is able to human like behaviors and reproduce them as such.

6.2.1 Overall Architecture of our work

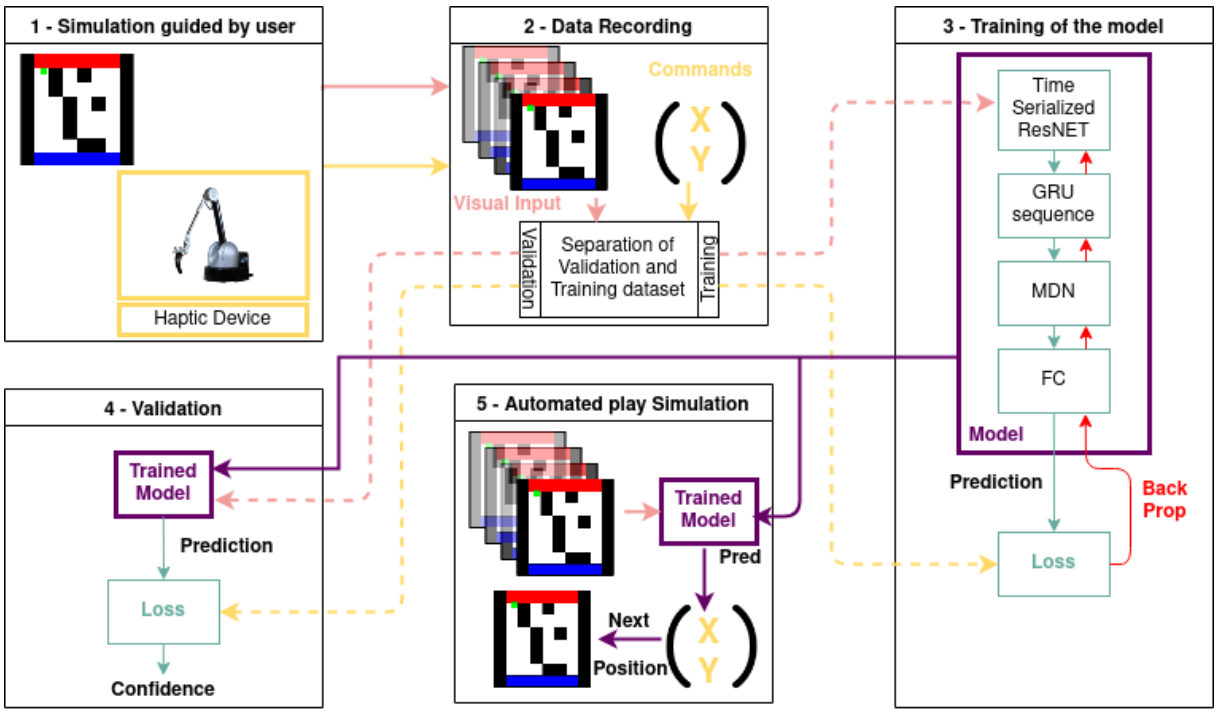


FIGURE 6.1 – *Overall architecture model training and testing*

Our work is composed of 5 main parts.
These parts are presented in the following figure (Fig. 6.1). We will describe each of the blocks contained in the figure.

- **Block 1 - Simulation guided by user**

  In order to create data to be learnt by our model, we asked a participant to control the robot among the haptic feedback device and to move within the randomly generated path up to the red color bar starting from the blue area 6.2. This participant will be called afterward our Human Expert. Once the red color bar reached a new path is generated and the blue and red rectangles have their position swapped. By doing this the participant can move as long as needed with no need of ever re-centering the haptic device neither stopping his motion. We decided to give a two Dimensional task to the participant, where the goal is to move a cursor freely in a video game like environment. [Moon and Seo, 2021, Poeller et al., 2018].
  By doing this we can easily obtain multiple kind of data : the visualization of the task that the participant is currently seeing on the screen and the motion that the participant is doing while controlling cursor at every time step.
  Motion data can be of many kind but we decided to focus on relative position motion in the task space.

  More details in section Sect. 6.3.

- **Block 2 - Data Recording** To get the visual and positional data needed, we recorded the picture in the simulator at a frequency of 12Hz. Indeed the more data is recorded the more time training will take so we decided that it was a good compromise between performance and training time. The recording of the position of the robot in the simulator frame is done as the same frequency and stored in form of a CSV file.

  Once all the trials were recorded, it is needed to transform the data in a format that fits our architecture. First images are uncompressed (due to the JPG compression) and down-scaled to 32x32 pixels and stacked to form batches of images representing a video clip of n images. The downscaling is a mandatory step needed for RESnet, which was pretrained on RGB 32x32pixels pictures. The shape of our only network input is (32,32,3,n) with n the number of previous images from current to minus n-1. We used n=5 during our experiments. This process is applied to every picture present in our image database, this is what we will call from now on our input dataset.

  Then every batch representing a video in the input dataset is attributed a label. This label corresponds to the position of the robot in the next picture. Its shape is (2,). This is what we will call the label dataset.

- **Block 3 - Training**

  During the training phase the 2D relative position from the participant's motion dataset is used in order to label each video clip input. Instead of using the current frame's 2D relative position we chose to use the next frame position in order to achieve prediction from the participant's motion. Training results can be found in Sect. 6.9.

- **Block 4 - Validation** The previously generated model is tested on 30 percent of the input dataset, the validation dataset, and loss is computed in order to get the validation loss.

- **Block 5 - Automated Play Simulation**

  Once trained, our model is integrated in the simulator and predicts the next (X,Y) position with images as input in real-time. This inferred position is then applied to the robot in order to close the loop of the control.

6.3 Visual Simulation

The graphic environment used for our robot simulation was created in Python with PyQT5. This simple interface gives only important information to the human participant like where the goal is, what are the obstacles and the position of the robot controlled by the participant.

Since our goal is to learn from motion our Deep Learning algorithm should use labels from the human motion dataset and since we want to imitate the human reaction to some pattern on the 2D task, we would need as input the recording of the visual task achieved by the participant.

Due to the temporal property of motion in general we needed to use time series as input for our network, hence we chose to use Gated Recurrent Unit (GRU) as it has been largely used in pHRI and robotics [Sarvadevabhatla et al., 2016, Luo et al., 2021, Yu et al., 2020]

By using haptic feedback device 2 points are achieved.

- First the human motion is not constrained due to the transparency and the low inertia of these devices. Since the interface length and width both fit inside haptic device workspace, the participant can move freely in the plane without constraints and without thinking about re-calibrating the physical motion in the simulation. We also choose the motion to be achieved in a horizontal plane, so we could neglect gravity effects.

- The second point is that thanks to the haptic device the participant feels forces when a collision occurs between the cursor and any obstacle. This has two consequences : the participant feels more embodied in the system and generate only data that corresponds well to the task. Indeed, simply clamping the robot out of the obstacles would have the haptic feedback frame and the simulator frame be de-synchronized.
  With haptic feedback, the motion is achieved by the operator, even if he was constrained, which reinforces the impression that obstacle should be avoided.
  Also not disynchronising the motion of the participant and the image rendered in the interface participate in the embodiment sensation of the tele-operator, indeed we used a large haptic interface which allowed our user to never have to think to

re-center the device inside the graphic interface's frame. This would have been necessary if we had used a little haptic device interface for example.

The communication of the haptic device and our graphical simulator is achieved with ROS (noetic). The haptic feedback device can be set up in impedance mode or admittance mode. For our use, the haptic feedback device is set up in impedance mode, which makes us able to subscribe to the haptic feedback physical position data stream and to publish forces to the device in order to simulate collisions, or impedance changes. With this mode the teleoperator guides the robot in simulation by changing his own hand position, and perceives collisions as the resultant forces that pushes his hand away.

6.4 Train pattern and test pattern

We decide to separate out training maze pattern and our model testing maze pattern for two main reasons:
Firstly, by having different patterns that the network was never taught on we can make sure that the model generalizes well beyond the training set.
Secondly the test mazes pattern are chosen in a way it's easy to analyse human motion and accordingly the model's predicted motion.

The simulator is composed of a 8x8 grid where the blue first row represents the departure and the red last row the arrival line while the first and last column entirely made out of black constraining the area.
In the middle, remains a white 6x6 grid. (Fig. 6.2)

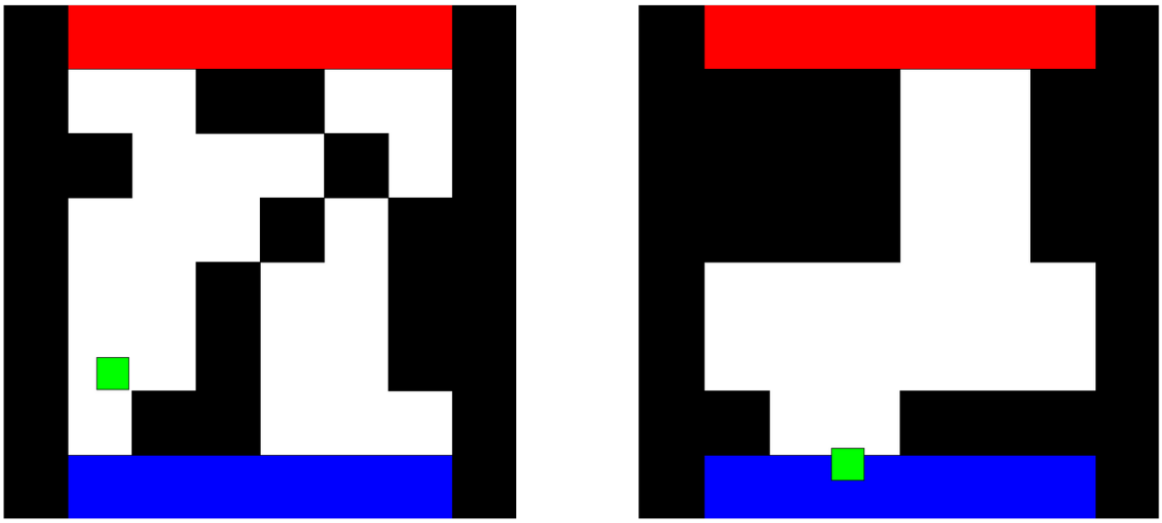


FIGURE 6.2 – *Left : Randomly generated path Right : path the model was never trained on*

6.5 Deep Learning Model

6.5.1 Pre-trained visual feature network : ResNET

Before talking about pre-tained convolutional network, there is the need to understand convolutional network and their uses in robotics. The use of computer vision increased in the last decades with the decrease of costs of both camera technologies and computing costs. Regular computer vision fields used a lot of convolutions to find region of interests or features in a pictures . In 2012 Ciresan and al [Cireşan et al., 2012], applied Convolutional Neural Networks (CNN) that learns convolution filters and parameters in order to recognize and classify handwritten numbers from MNIST, a handwritten number database.

Kaiming He and al. introduced RESnet in 2015 [He et al., 2016] and it has been one of the biggest advances in Deep Learning : how to stack hundreds of layers in a deep model without having a problem of degradation of the convergence result neither the vanishing/exploding gradient problems. One part of the answer lies in the particularity of RESnet (and HighWayNet a few months before). The residual elements, a feed-forward connection that "skips" 2 layers every 2 layers, that tends to avoid a specific layer if this layer hurt the performances of the complete network 6.3. ResNet was pre-trained on ImageNet, a database composed of 14 Million images describing more than 20 000 categories of real life objects. It is composed of roughly 27 Millions weights without counting the top layer used when classifying from different category. Hence our choice to use this particular network : its efficiency in real life object detection would allow us to use it to detect objects from a simulator as well as from real objects, with all the complication that brings. RESnet is the most cited neural network paper, in 2022.

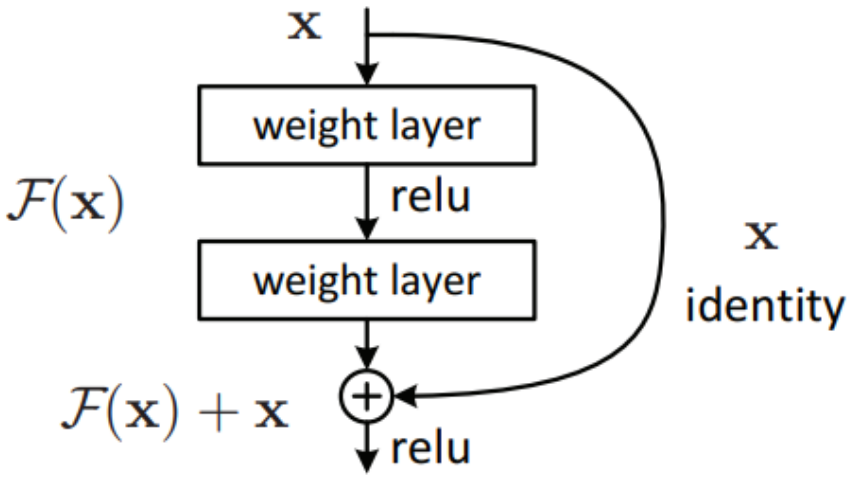


FIGURE 6.3 – *A residual block : x represent the input, F a layer of any type. The output is then F(x)+x which make possible the fact to avoid F(x) during the back-propagation, Source He et al. [2016]*

6.5.2 Learning temporal structure : Gated Recurrent Units

As explained in Sect. 6.2.1, we want to learn motion, thus we need an architecture that can learn temporality, not only visual features, like Recurrent Neural Networks, Long Short Term Memories and Gated Recurrent Units.

For our architecture we chose to use Gated Recurrent Unit (GRU) because they are the more recent and started to be used in robotics too [Chen et al., 2022, Patel et al.]. Indeed, GRU are preferred over LSTM in case of smaller dataset. LSTM have a more complex architecture compared to GRU, hence they tend to need more data to be trained, which is not generaly the case in LfD.

6.5.3 Mixture Density Network

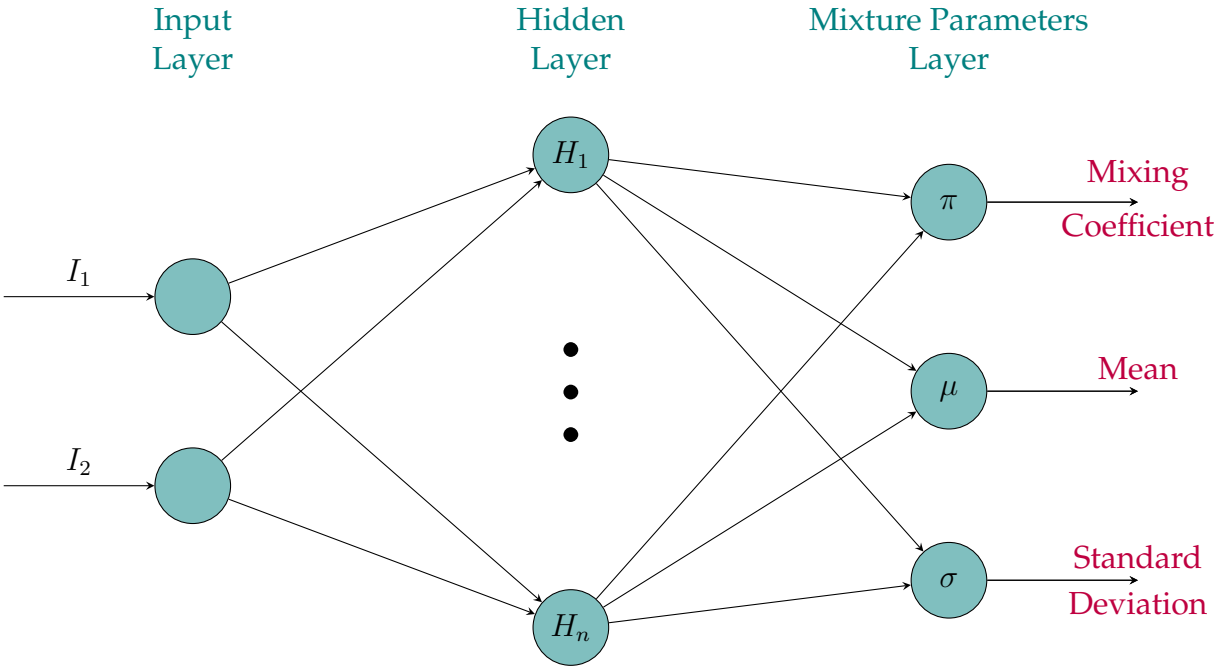


FIGURE 6.4 – *Example of a MDN architecture with 2 Inputs I_1 and I_2, n Hidden Layers and the π,μ,α outputs.*

The Mixture Density Networks (MDN) were first presented by Christopher M Bishop in 1994 [Bishop, 1994]. They were developed in order to have a Neural Network output a probability distribution when given some input tensor instead of outputting a simple tensor regressed from the input.

This allows our network to output not a single tensor but a distribution characterized by one or multiple parameters based on Gaussian Mixture Model (GMM)

Indeed Mixture Density Networks are widely used in the more or less recent Deep Learning field, for example it is being used by Apple in the Siri's technology to find means and variances of speech features for speech recognition [Siri, 2017], for speech synthesis [Zen and Senior, 2014], for Uncertainty aware learning from complex real-

world driving [Choi et al., 2018], for Movement Primitive Learning [Zhou et al., 2020], for robot movement self recognition in a mirror [Lanillos, 2020].

Let's take the example of a Gaussian distribution $G(x)$, in this case these parameters are : μ the conditional mean and σ the conditional standard deviation

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}\right] \tag{6.1}$$

A Gaussian Mixture Model $p(y|x)$ is represented as a sum of Gaussian distributions $G(x)$ multiplied by $\pi_i(x)$ the mixing coefficients, which corresponds to the number of mixture layers added after the hidden layers. C denotes the number of Gaussians used in the GMM. It is important to note that the sum of the $\pi_i(x)$ mixing coefficients is equal to 1.

$$p(y|x) = \sum_{c=1}^{C} \pi_c(x) G(x, \sigma, \mu), \sum_{c=1}^{C} \pi_c(x) = 1 \tag{6.2}$$

GMM are able to represent probabilities that are not only uni-modal but multi-modal with multiple means and variances.

A Mixture Density Network is a network that learns the π, μ, σ parameters of a GMM to represent an output; in our case, the output of our network.

$$\begin{bmatrix} (\mu_x, \mu_y)_1 & \dots & (\mu_x, \mu_y)_C \\ (\sigma_x, \sigma_y)_1 & \dots & (\sigma_x, \sigma_y)_C \\ \pi_1 & \dots & \pi_C \end{bmatrix}$$

In order to choose which Gaussian $G(x)$ from the GMM to use, the Softmax function $s(x)$ is used to get the maximal $\pi_c(x)$ value. Generally, the Softmax $s(x)$ value is used in Machine Learning as an activation function to get probabilities distribution from non-normalized real numbers outputs, then find the highest probability to determine the output of the network.

$$s(x) = \frac{\exp(h_i)}{\sum_{j=1}^{C} \exp(h_j)} \tag{6.3}$$

By using MDN and their probabilistic nature the output of our regression model is a density probability function, not a simple tensor. The usual used Mean Squared Error (MSE) cannot be used as is. It is generally admitted that instead of MSE the cost function used should be the Negative Log Likelihood (NLL) $L(x)$. But compared to MSE whose mathematical limit $\lim_{x\to+\infty} MSE(x)$ in 0, which is very useful in machine learning to estimate the results of the model, the Negative Log Likelihood has a mathematical limit $\lim_{x\to+\infty} NLL(x)$ not defined, that can also have negative results.

$$L(x) = -\ln \sum_{i=1}^{C} p(y|x) \tag{6.4}$$

which gives when the Negative Log Likelihood $L(x)$ is applied to our Gaussian Mixture Model :

$$L(p(y|x)) = \sum_{c=1}^{C} \pi_c(x) \left[\ln(\sigma) - \ln(2\pi) - \frac{(x-\mu)^2}{2\sigma^2} \right] \tag{6.5}$$

To summarize, a trained Mixture Density Network is able to predict a conditional density function of the labeled data from the input data. By using this conditional density function, the aim is to obtain an uncertainty or a probability about the accuracy of the output σ and the probability centered in μ, to compare a classic regression would have output only μ.

Moreover by using multiples mixtures components π, the output is containing C different Gaussian Distributions which we have to choose from by using the Softmax. Overall it gives more detail about the uncertainty of the network output, which is something important in robotics in order to have reliable prediction would it be movement generation or haptic guidance.

For the pre-built implementation of the Mixture Density Networks in TensorFlow, we used an available open-source implementation keras-mdn-layer [1].

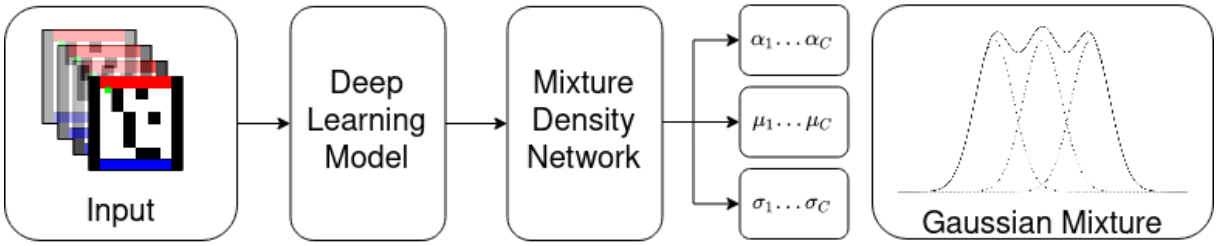


FIGURE 6.5 – *MDN layer inserted in our deep learning architecture, with the GMM output*

1. https://github.com/cpmpercussion/keras-mdn-layer

6.6 Training Data

The training has been done in 2 phases with 2 different datasets. First, we trained the network in order to get the model "understand" from the images the following points :

- what is the goal
- what can be moved
- where to move
- how to move

The second training is a fine-tuning training. The method used is called transfer learning, which consists in loading an already trained network and retrain it with another dataset. Since the network was already trained, fine tuning is used here, hence the Learning Rate LR is decreased from 0.0001 to 0.00001. The particularity of these datasets is to be able to represent different human motion patterns that are unique to R. Thanks to our haptic device, we use different physical constraints to have the human move differently but with different behavior, we will then acquire 3 datasets for transfer learning.

Our first dataset is composed of 38551 points which represents 2h of human motion without constraints made by our Human Expert. The (Fig. 6.6) represents the density of of the global dataset motion in the 2D task space The density of points is contained in the 800*800 pixel, size of our maze in the simulator.

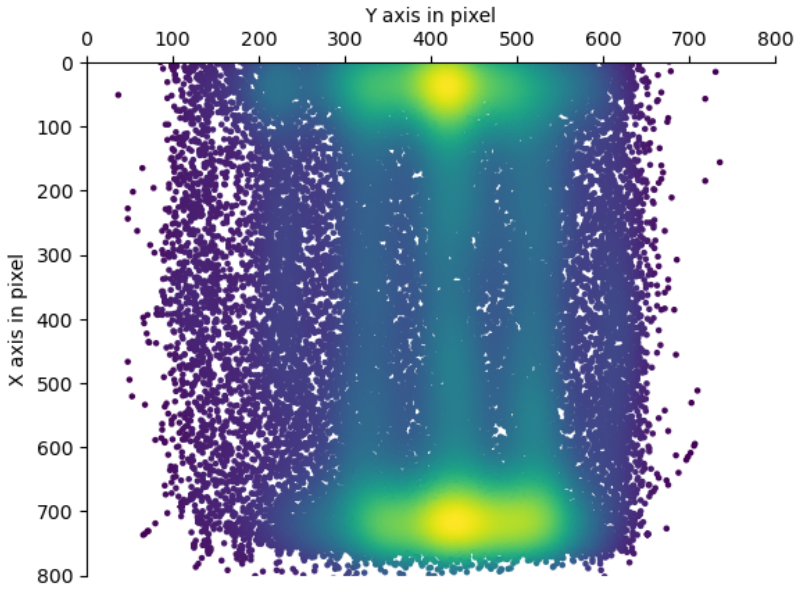


FIGURE 6.6 – *Representation in position of the pretraining dataset. This represents all the position accessed by our Human Expert during all of the dataset creation phase. The density of point is represented by the color from blue to yellow, with yellow representing a high density and blue a low density. This figure shows that the whole area was accessed equally by our Human Expert,except from the middle of the "start" zone and the "finish" zone.*

Afterwards, we used a second dataset. In this new dataset the only change applied is a 15N.m-1.s damping constraint added on the Y axis of the haptic feedback device. As a result, when the human operator will move left or right, a force in the opposite direction relative to its velocity will be applied. The human will feel it as a viscosity that tries to slow down the motion. In term of motion its possible to understand what kind of effect these changes will have on the motion but as the human in the loop, we do not exactly know how the whole trajectory will be affected.

The fine-tuning dataset procedure was done 3 times, one for each conditions to get different dynamics.

For the example, one of our three fine tuning dataset is composed of 23371 points which represents 1h of human motion with a Y-axis damping constraint The figure (Fig. 6.7) represents the density of the global dataset motion in the 2D task space

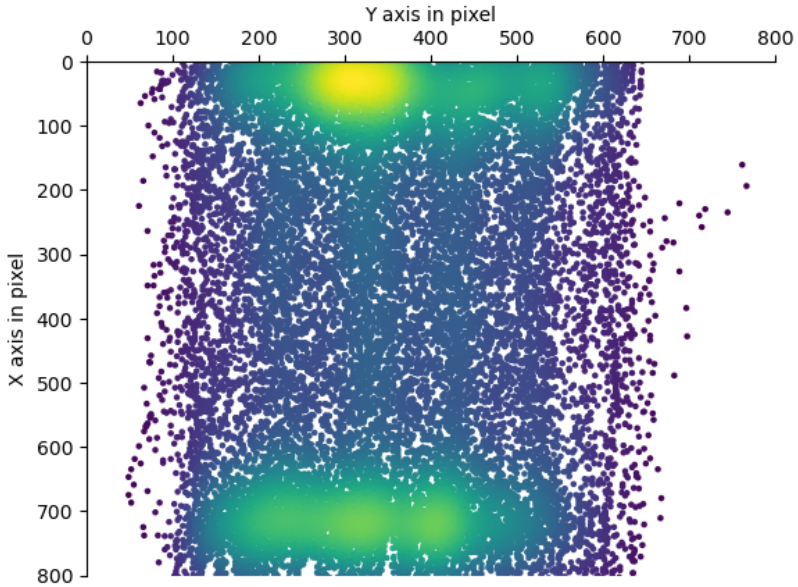


FIGURE 6.7 – *Representation in position of the fine tuning dataset with Y axis damping constraints. This represents all the position accessed by our Human Expert during all of the dataset creation phase. The density of point is represented by the color from blue to yellow, with yellow representing a high Density and blue a low one. This figure shows that the whole area was accessed equally by our Human Expert, except from the middle of the "start" zone and the "finish" zone.*

In order to get different behavior in terms of dynamics, a total of 3 dataset were created :

- One with a $F_x = 15N.m^{-1}.s$ damping constraint applied on the Y axis (Right-Left) of the haptic feedback device.

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = \begin{bmatrix} 0 & 15 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \tag{6.6}$$

- One with a $F_x = 15N.m^{-1}.s$ damping constraint applied on the X (Front-Back) axis of the haptic feedback device.

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = \begin{bmatrix} 15 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \tag{6.7}$$

- One with $F_x = 0N.m^{-1}.s$ damping constraints.

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \tag{6.8}$$

with $\dot{x}$ and $\dot{y}$ the velocities in the Picture frame

6.7 First Approach

Our first approach to the problem when we started was to decouple the image recognition part and the motion trajectory learning. We chose an architecture that would be made of two blocs, one based on YOLO or a pre-trained network that outputs visual features as 2D positions of multiples objects, and a RNN bloc to analyse these objects motion. But we realized a mistake while testing this architecture, reducing the dimension of a picture to the simple position to the object does not help the network to train, the features of the first bloc are too rich of information. These positions, while easily understandable by any human, seem not to be so easy for an untrained network.
We then decide to move to a close architecture at the difference that the output of the first bloc would be a visual feature tensor from ResNet. Hence it cannot be understood by an human, it seemed to be easier for the RNN part to learn from this. Moreover, it is still possible to visualize ResNet layers to try to "understand" what is happening, and check the visual data integrity is still present in the network after the data-processing phases.

6.8 Model Architecture

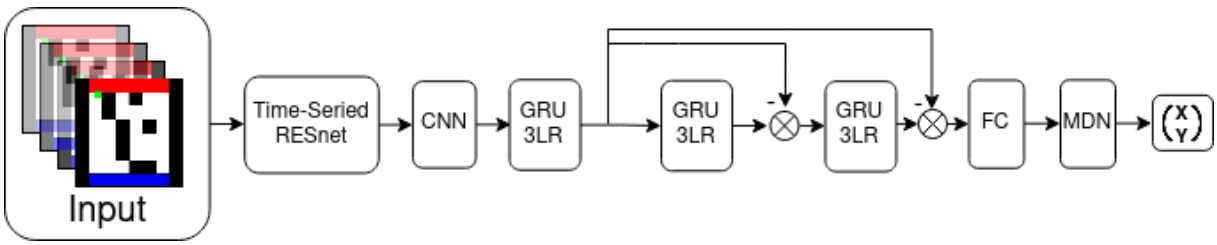


FIGURE 6.8 – *Our end-to-end deep learning based neural network architecture*

In the previous section we defined every kind of architecture and explained our choice to use them. In this section we will define the global architecture of our model. This architecture is presented 6.8

As explained in Sect. 6.2.1 the input of our network is an array of RGB pictures, a video clip.
In order to apply the pretrained RESnet network to a video, it is mandatory to use what is called a time-series in TensorFlow in order to apply RESnet to every frame of the input. In order to reduce the dimension of these features, a CNN is applied through a time-serie to each of RESnet output. Once done, our video clip is transformed in a timed array of visual features.
From now we will use Gated Recurrent Unit since we explained before it has been showed they can grasp temporality.
We define in (Fig. 6.8) a GRU 3LR which represent three GRU layers stacked with their output linked in the input of the next one. It has been showed to improve the efficiency of RNN [Reimers and Gurevych, 2017, Wu et al., 2016, He et al., 2017] Inspired by the concept of residual blocks, we propose to have a similar arrangement of our GRU 3LR layers. This would avoid using a GRU if its result in training is not good enough. Then a Fully Connected layer to reduce the dimension of the output followed by the MDN layer. An example of TensorFlow model defining our architecture can be found in appendix:tfmodelcode All the units present in each layer are detailed in Appendix 8

6.9 Model Training

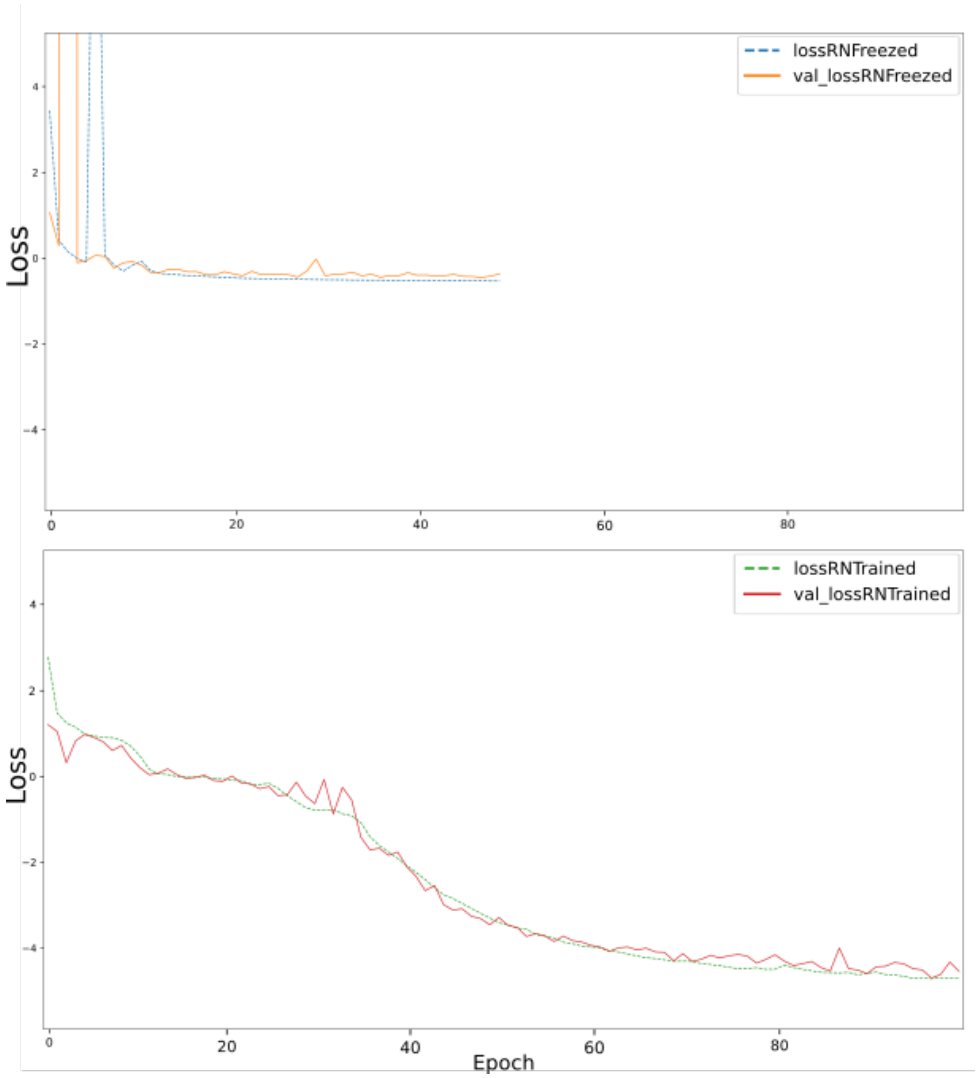


FIGURE 6.9 – *First Training Dataset : Loss and Validation Loss over epochs. The upper graph represent the 50 first epochs with ResNet layers freezed. The lower graph represents the 100 epochs with all weight unfreezed.*

When using architecture containing Pre-trained models, it is generally admitted that during the first training epochs the Pre-trained model's weight should be freezed. Indeed, since the other layers's weights are set-up randomly, the back-propagation would un-optimize the weight of the pre-trained layers.

On the training loss curve we will have the loss and the validation loss, as expected in most training, but we will also have the loss and the validation loss while the pre-trained layers were freezed, in our case, the ResNet internal layers (Fig. 6.9).

Once the model trained, we fine-tuned this same model with no constraint (Fig. 6.10), Y-axis constrained data (Fig. 6.11) and X-axis constrained data (Fig. 6.12). This training phase is done with ResNet freezed, since we want only the later layers to be fine-tuned.

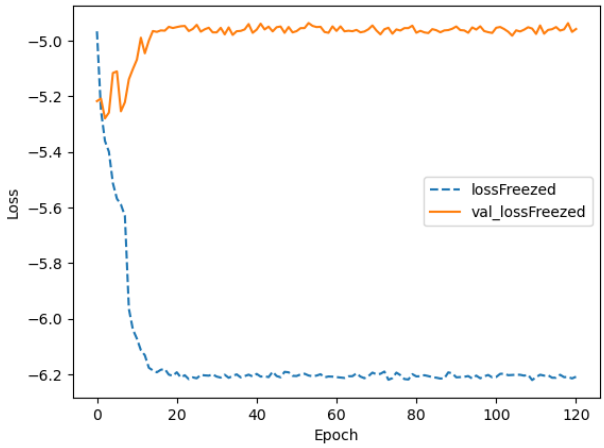


FIGURE 6.10 – *Fine Tuning Training Loss with no constraints dataset. Since nothing new was added to this dataset the network did not learn much more, hence the loss and validation loss diverged from the very beginning*

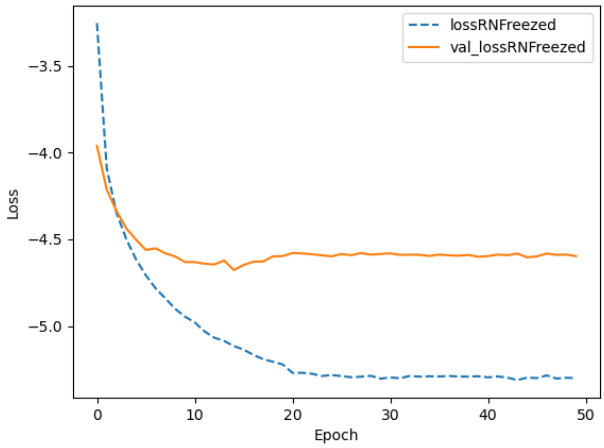


FIGURE 6.11 – *Fine Tuning Training Loss with Right Left constraints dataset*

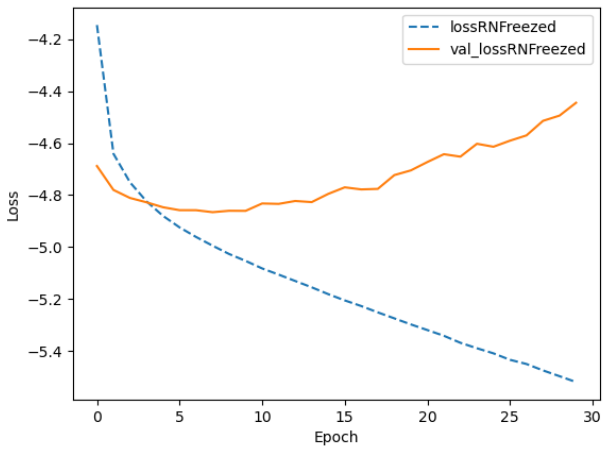


FIGURE 6.12 – *Fine Tuning Training Loss with Up Down constraints dataset*

6.10 RESnet internal layers

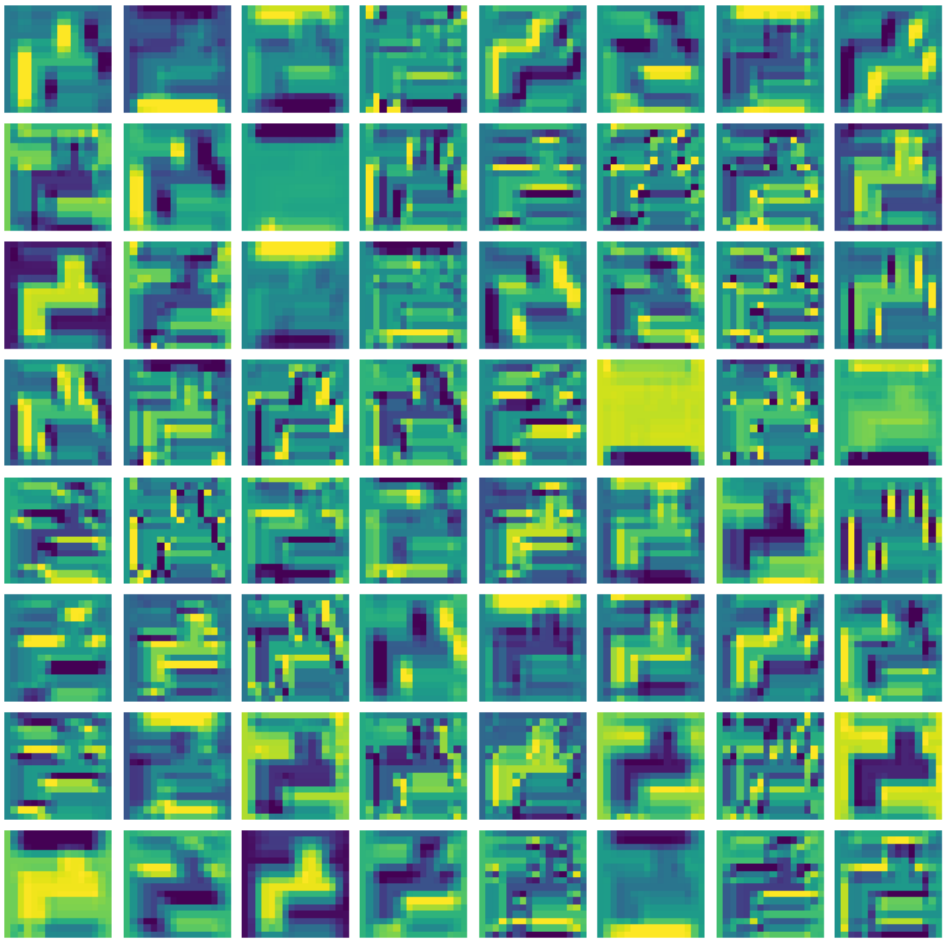


FIGURE 6.13 – *This represents 64 different Convolutions applied to the input image of RestNet at the internal Layer of Convolution number 2. It is possible to understand why be used by the network to output it's feature vector because some convolutions makes sense for an human.*

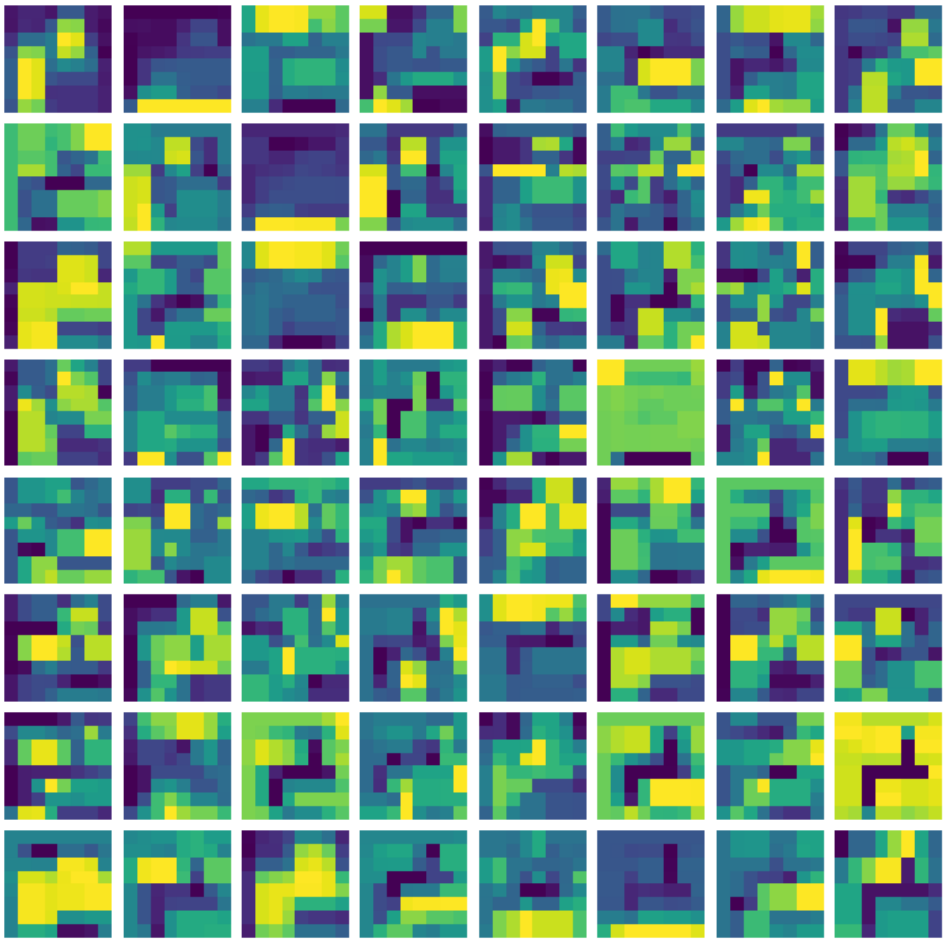


FIGURE 6.14 – *This represents 64 different Convolutions applied to the input image of RestNet at the internal Layer of Convolution number 4.*

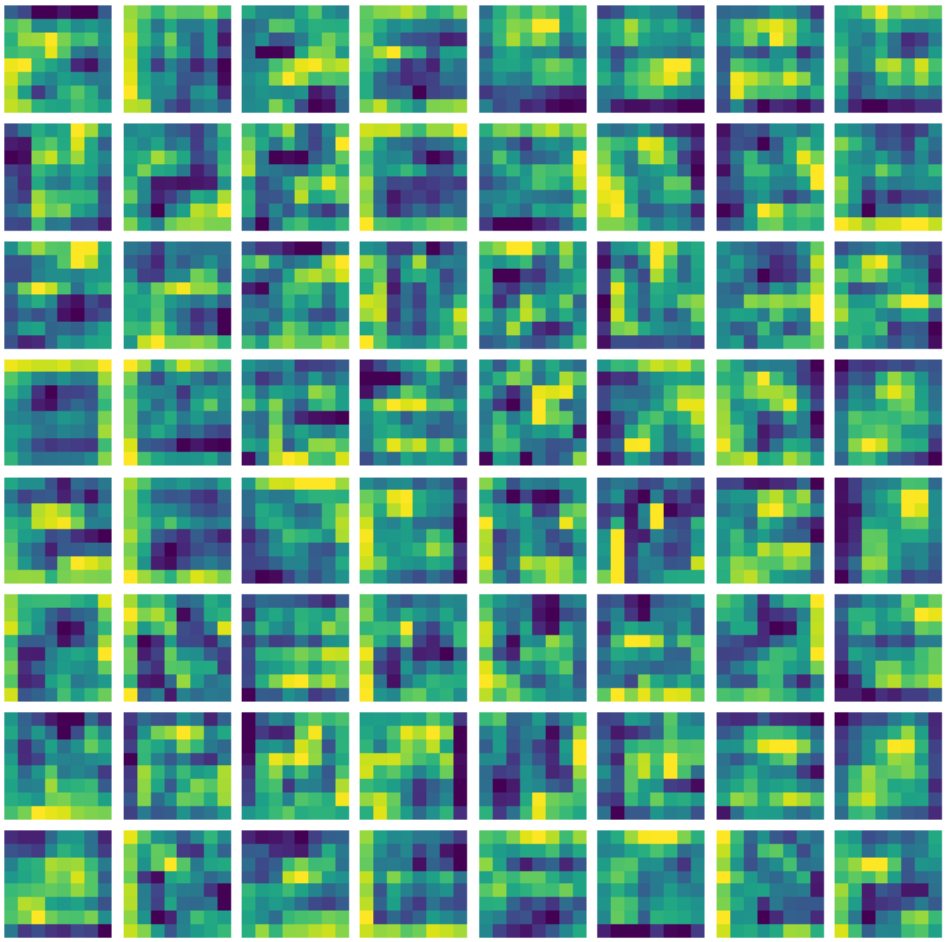


FIGURE 6.15 – *This represents 64 different Convolutions applied to the input image of RestNet at the internal Layer of Convolution number 11. This layer become hard to analyse for an human but is closer to the output of the ResNet output feature vector*

As explained before we use ResNet, a pre-trained visual recognition network. In ResNet it is possible to get the output of internal middle layers so that when applied an input picture (Fig. 6.13), (Fig. 6.14) and (Fig. 6.15) the output of the nb 2, nb 4 and nb 11 internal layers. This is especially useful for debugging, in order to be sure that all data input in the system are well input, uncompressed, etc. It also represents CNN layers, so some layers have have specifically trained to detect wall, some to detect the path, some to detect the start and finish area, etc. By comparing output from different input picture we are also able to check which convolutions are dependant of the differences between our input and which are not. But the deeper the layer we want to visualize,

the more difficult it is to interpret (Fig. 6.15).

6.11 Agent Task Achievement Analysis

In this section we will present the results obtained with the Deep Learning Agent that was described in the previous chapter. We will compare our Agent motion and behavior with the Human Expert that trained the datasets and with our participants in order to prove that our Agent behavior can be treated as human-like in our scenario. Our robot simulation and our scenarios were defined in the precedent Section 6.3

As described before, our robot is simulated by a green square moving in a white path. The goal of the Human Expert was to go from the blue "starting" area to the red "finish" area while avoiding entering black blocks.

All the Agent training was done trough LfD. Our Agent had no preconceived knowledge, neither policies, describing the environment, motions, dynamics or the task to accomplish.

In 6.11 which link to an online video, we show a sample of our Human Expert motion dataset, while moving the green robot in a few randomly generated maze, recorded with our simulator and haptic feedback device.

FIGURE 6.16 – *Video of our Human Expert dataset used for training (the QR code is a clickable link)*

Our Agent learnt a few properties inherent to the task (Fig. 6.18), (Fig. 6.19), (Fig. 6.20) and a few behavior of our human expert teacher (Fig. 6.17), (Fig. 6.21).

The next figures represent time sequence motions that have to be read from Left to Right, following the numbers. They describe properties learnt by our agent.

6.11.1 Agent obstacle avoidance motion

In Figure 6.17, we generated a path not contained in the training dataset, fixed the robot in the blue "starting" area next to the path beginning. As a result our Agent avoided the obstacles and found a trajectory to get to the red "finish" area without colliding in the black "wall" area.

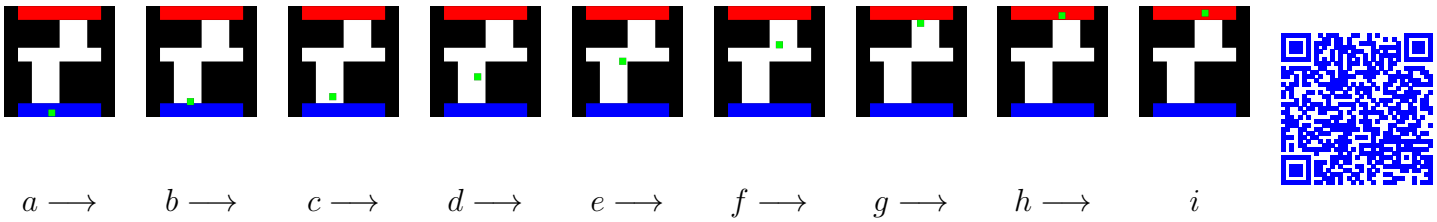


FIGURE 6.17 – *Fixed path that is not contained in our training dataset. The robot is placed in the blue starting area. The Agent finds a path to move the green robot to the red "finish" area while avoiding the obstacle.*

In Figure 6.18 we chose a path that never appeared in our training dataset and we swapped the blue "start" area and the red "finish" area from the frame 6.18.e. We wanted to see if our Agent could change direction while moving, even if this was never done in the training dataset. As a result we observe that our Agent learnt direction based on the color area but also learnt to change direction during a motion, not only when the robot was in the starting area.

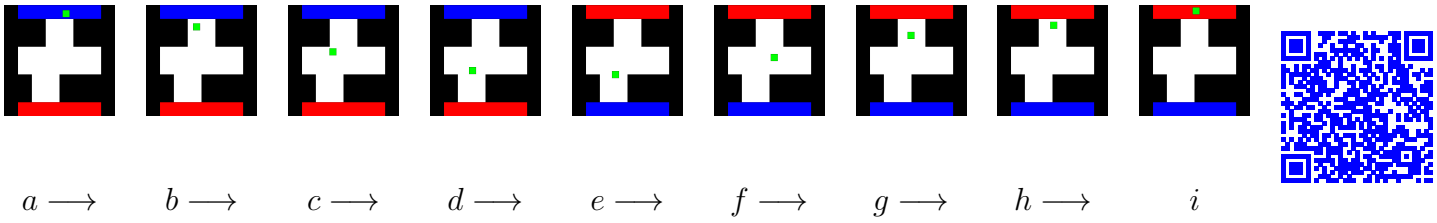


FIGURE 6.18 – *Red and Blue area swap places. It meant for the Human to change direction during training since he was asked to go toward the red colored "finish" area. Our Agent also learnt to change direction without never being explicitly explained.*

For the 2 next Figures, we choose a randomly generated mazes and put the robot in the blue "start" area in 2 different positions. In the Figure 6.19, the robot is placed left and in the Figure 6.20 the robot is placed right. We observed that in both picture the Agent find a way to the red "finish" area by taking completely different trajectories. Note that this behavior shows that our Agent generalized movement according to the color of the finish area and not its physical position.

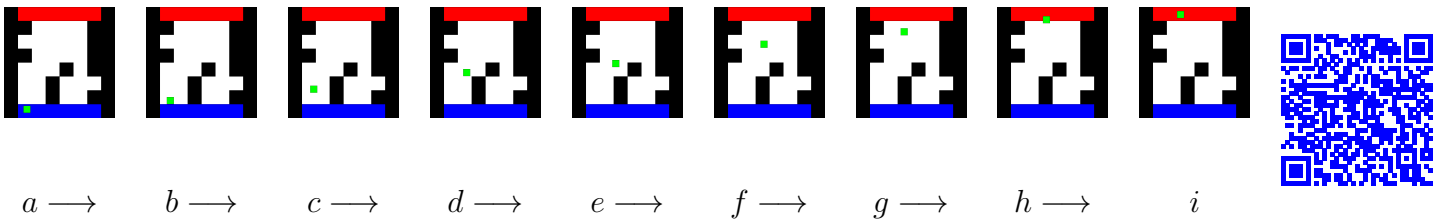


FIGURE 6.19 – *Example of a randomly generated maze, not contained in our training dataset. Our robot is placed in the left part of blue starting area. The agent find a path to go to the red "finish" area*

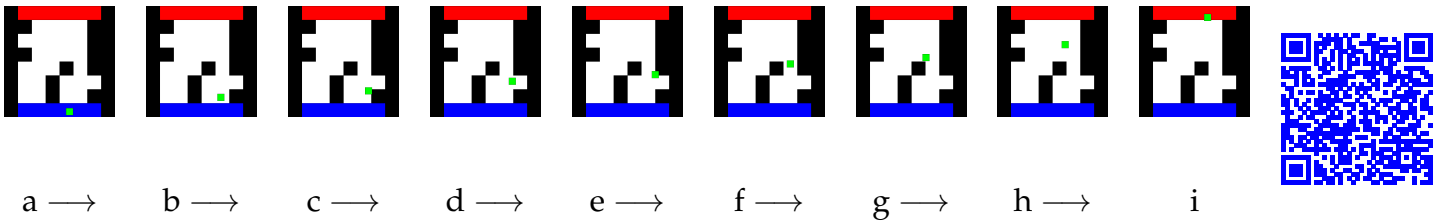


FIGURE 6.20 – *Same randomly generated maze as in (Fig. 6.19). This time our robot is placed in the right part of blue starting area. The agent find a path to go to the red "finish" area. The fact that the agent did not tele-ported the robot to take the same path as the first example shows that our Agent has generalized.*

6.11.2 Agent motion example in case the path is not found by the Agent

In Figure 6.21, we purposely generated a random maze where our robot could get stuck and place it purposely in front of the path where it could get stuck. We observe that our Agent learnt to go in the right direction despite the fact that the path is a dead-end but learnt to come back to the blue "start" area when it got stuck with no valid way to move on anymore. This is a good example of generalization because this phenomenon represented only a small number of cases in the training dataset.

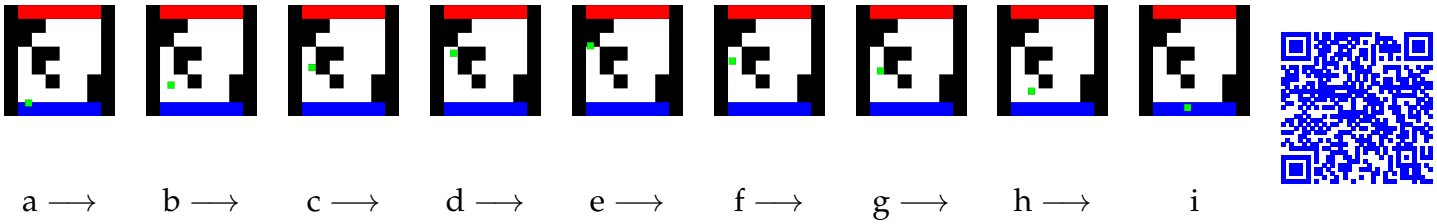


FIGURE 6.21 – *Example of another randomly generated maze, not contained in our training dataset. The robot is placed in the blue starting area, in front of an obstacle. The agent does not find a path to go to the red "finish" area so came back in the blue "starting" area*

6.12 Comparison with human movement

In the previous section, our Agent has been capable of moving the robot in the maze, avoiding obstacles and learned to reach the red finish area even when the physical positions changed, after the training phases by our human expert. In this section we will analyze our agent resulting motion. In order to do this we will use a fixed maze with only one possible way to move through. This allows us to make statistics on the data of multiple human participants and our Agent.

We asked 7 participants to perform in 3 mazes that were not contained in training datasets we selected with the help of our haptic feedback device. We also asked our human expert to perform in the same 3 mazes.

Each participant was asked to move in the maze in the order as they appeared, while going from the blue starting area to the red finish area while staying as much as possible in the path of the maze and avoid touching the black area. We did not specify any speed or time to our participants and if they touched the black wall, they felt an haptic response to have they go back outside of the walls. We chose 3 dynamics constraints 6.6 and asked them to perform the aforementioned task for every condition, in different order for each participants. We recorded the position and the speed of the robot during all the participants trials.

Our participants were shon the same 3 dynamics constraints 6.6 used for the creation of the dataset used in the transfer learning phase of our Agent, as explained in last Chapter.

By analyzing the position of our Agent and our participant in each case we obtained the following results (Fig. 6.22). In orange is plotted the inter participant mean trajectory, in light orange their standard deviation, and in cyan our Agent trajectory.

For our 9 cases, 3 mazes and 3 constraints, we computed the Pearson Coefficient between the Agent motion and inter-participant mean, in order to quantify how much both trajectories matches. We get for all of our 9 cases a Pearson Coefficient > 0.79 hence a P value inferior to 0.001.

We can see that our Agent has learnt to make human-like trajectories, even in mazes clearly different from the training dataset. In most of the trials, our Agent did not collide with the black wall and could go up to the red finish area.
In the next section we will have a look at velocities of our agent in the maze. Indeed our agent was never labeled with velocities during the training phase, neither taught any notion of speed.

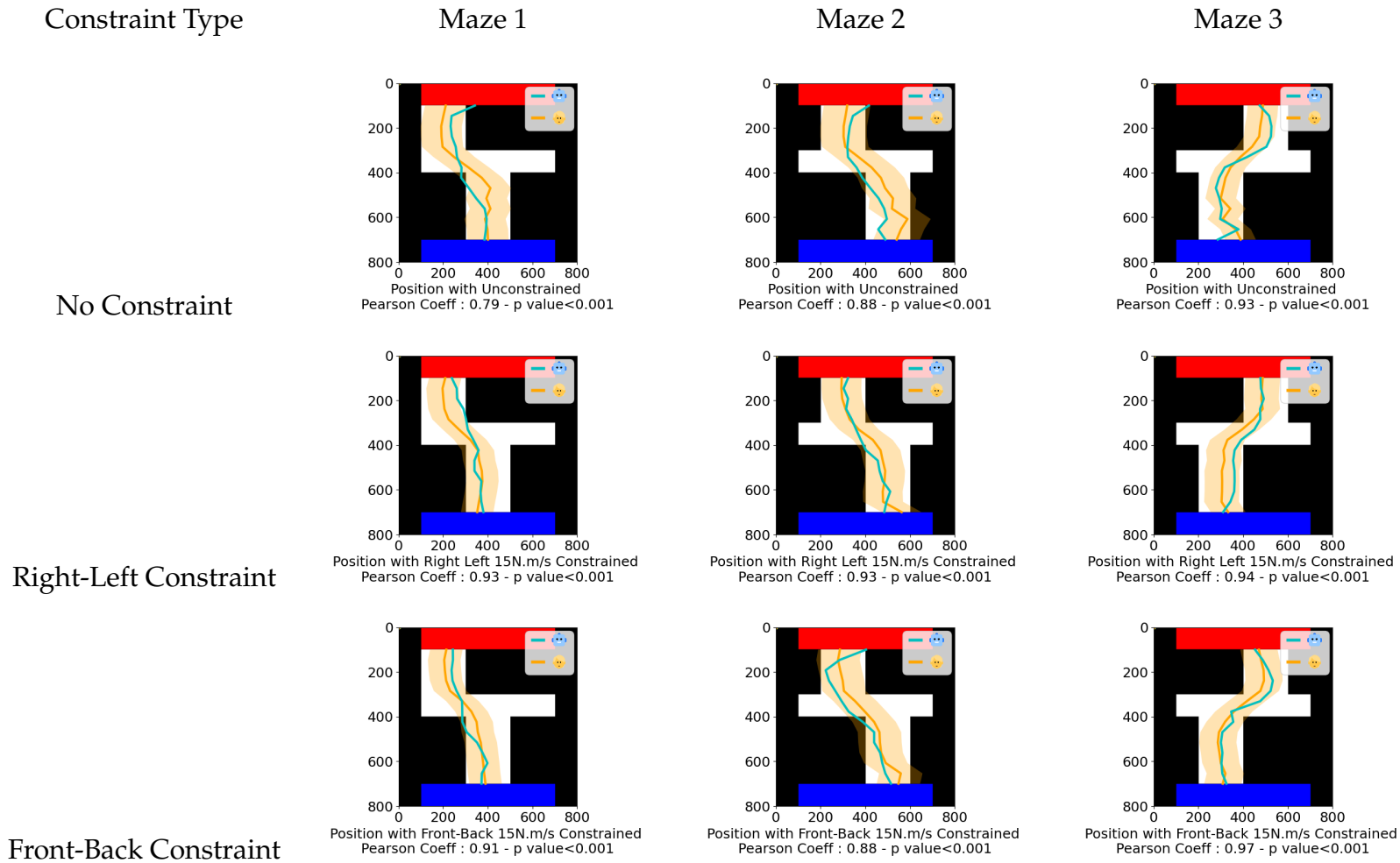


FIGURE 6.22 – *Participants and Agent trajectories :*
For our 9 cases, 3 mazes and 3 constraints, in Orange is represented the inter-participant mean of our 7 participants, in shaded orange its standard deviation and in cyan the Agent motion. For every maze we computed the Pearson Coefficient between both Agent motion and inter-participant mean in order to quantify how much both trajectories matches. We get for all of our 9 cases a Pearson value inferior to 0.001

6.13 Comparison with human velocity

In the previous section we showed that our Agent had learnt to make human like trajectories, we analyzed the velocities of our Agent. We found they were also matching and decided to use the velocities value to look at two points :

- Does our Agent matches human's velocity in each of the chosen maze?
- Is human behavior changing between our constraint cases and has our Agent learn these changes?

In order to verify our first question we used the data collected in the experiment detailed in , and chose for multiple reasons to use velocity data. Indeed velocities are less constrained by the maze shape than the positions, so diverging results could happen naturally. Any agent that learnt how to move in this maze would give such trajectory results, because in order to succeed there are no other choices than to take this path.

We noticed some differences occurred between the velocities of our Human Expert and the participants. The first was the average velocity that was higher for the expert than for the other participants. We had to multiply our Agent maximum velocity by 0.89 in order to have realistic matches between the participant and our Agent velocities. The coefficient was computed from the ratio of the inter-participant mean and our Human Expert mean throughout every maze and every conditions. These can be explained by the fact that our Human Expert was overtrained on our task (the basic one and the three constraint related ones) for about 45 minutes each.

The second difference was related to the three dynamic constraints we chose. Indeed these dynamic constraints brought some differences in the behavior of our Human Expert. These differences were also visible in our Agent. But unfortunately these differences were not present in our participant. The particularity may also come from the participants that used the device a short time so the damping constraints were not strong enough to have them feel fatigue in order to change the way they move. We are not sure what caused this particular divergence in the results.

We obtained the following results by comparing Right-Left velocities between our Agent and our participants . We used Pearson coefficient to show that our agent velocities matches our inter-participant average velocities in each of our maze and case. We found that in 7 cases out of 9, the Pearson Coefficient were significant for the number of bin used in our data (14 bins were used). To be more precise, 1 case (PCoeff=0.48) has a P value inferior to 0.1, 1 of 0.05 (PCoeff=0.56), 4 inferior to 0.01 and 1 inferior to 0.001 (PCoeff=0.91). We can conclude that our Agent learnt some way close to "human-way" to move in these mazes. Now we have to see how it scores when our Agent is faced to dynamic changes.

In order to show that our different dynamics constraints were affecting the agent, we compared our Agent speed with the participant panel one. But we noticed that they were not matching, indeed, our Agent was trained by our Human Expert only, so it can reproduce only the way our Human Expert behaved. So we decided to compared our human expert velocities with our agent not with the participants.

In the (Fig. 6.24) we notice that between our three dynamic constraints there are effects on Front-Bottom velocity. We notice that the speed in the *Front-Back constrained* condition is significantly decreased on both Agent and Human Expert compared to the *Unconstrained* condition.
We also notice in 2 cases out of 3 that the *Right-Left constrained* condition's speed is higher on both Human Expert and the Agent compared to the *Unconstrained* condition. This means our Agent was able to learn details about dynamic changes during our fine-tuning that were only constituted of changes in term of damping constraints.

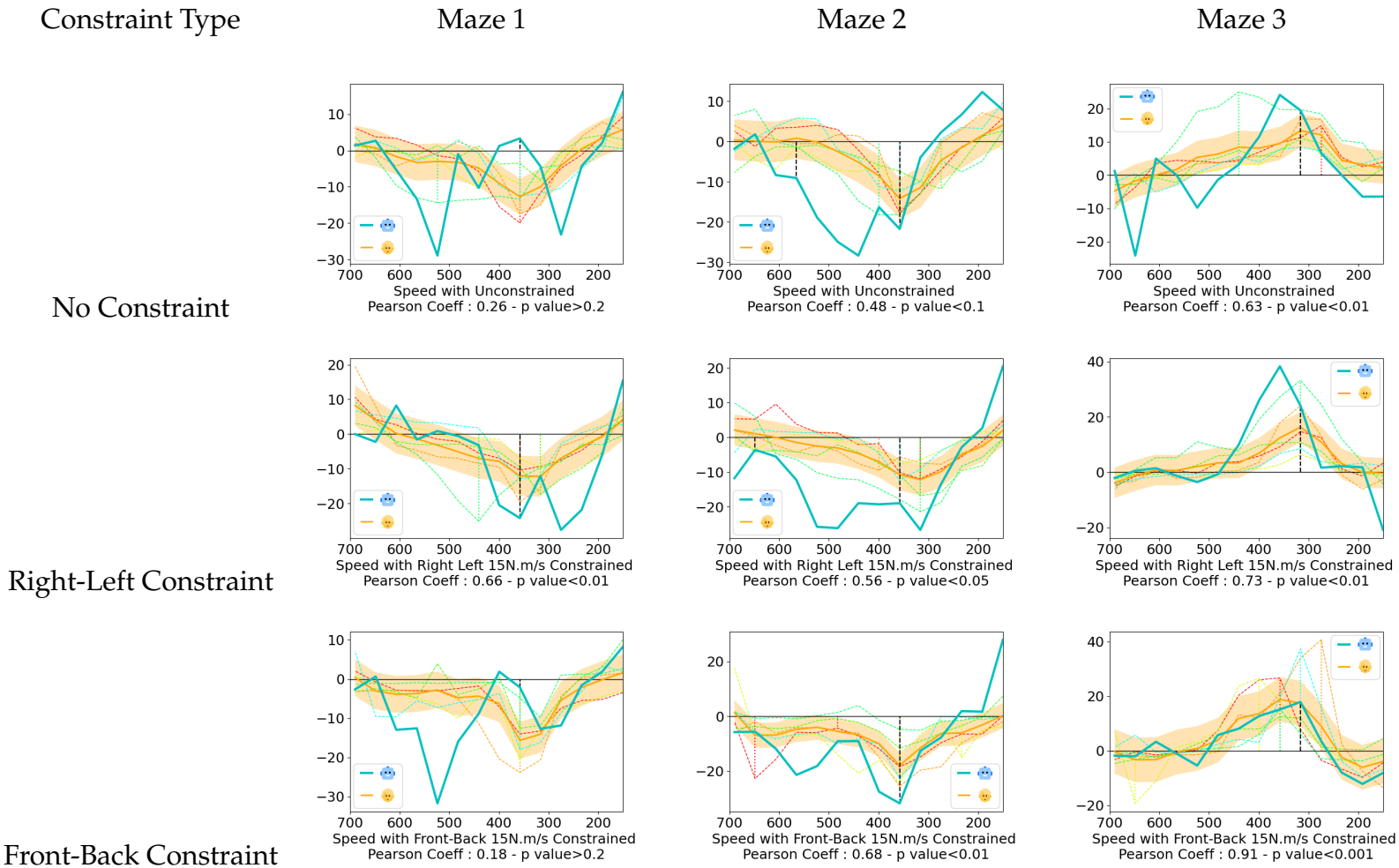


FIGURE 6.23 – *Right-Left Speed : Human Novice participant versus Agent*
In this figure we compare our participants' Right-Left axis (in orange) intra-participant average speed versus the trained Agent one (in cyan) for each of our 3 maze and our 3 conditions.
We used Pearson coefficient to compare the matching between each of the 2 averages. In our total of 9 cases, only 2 are insignificantly correlated in the Maze 1, 1 case has a Pearson value inferior to 0.1, 1 of 0.05, 4 inferior to 0.01 and 1 inferior to 0.001

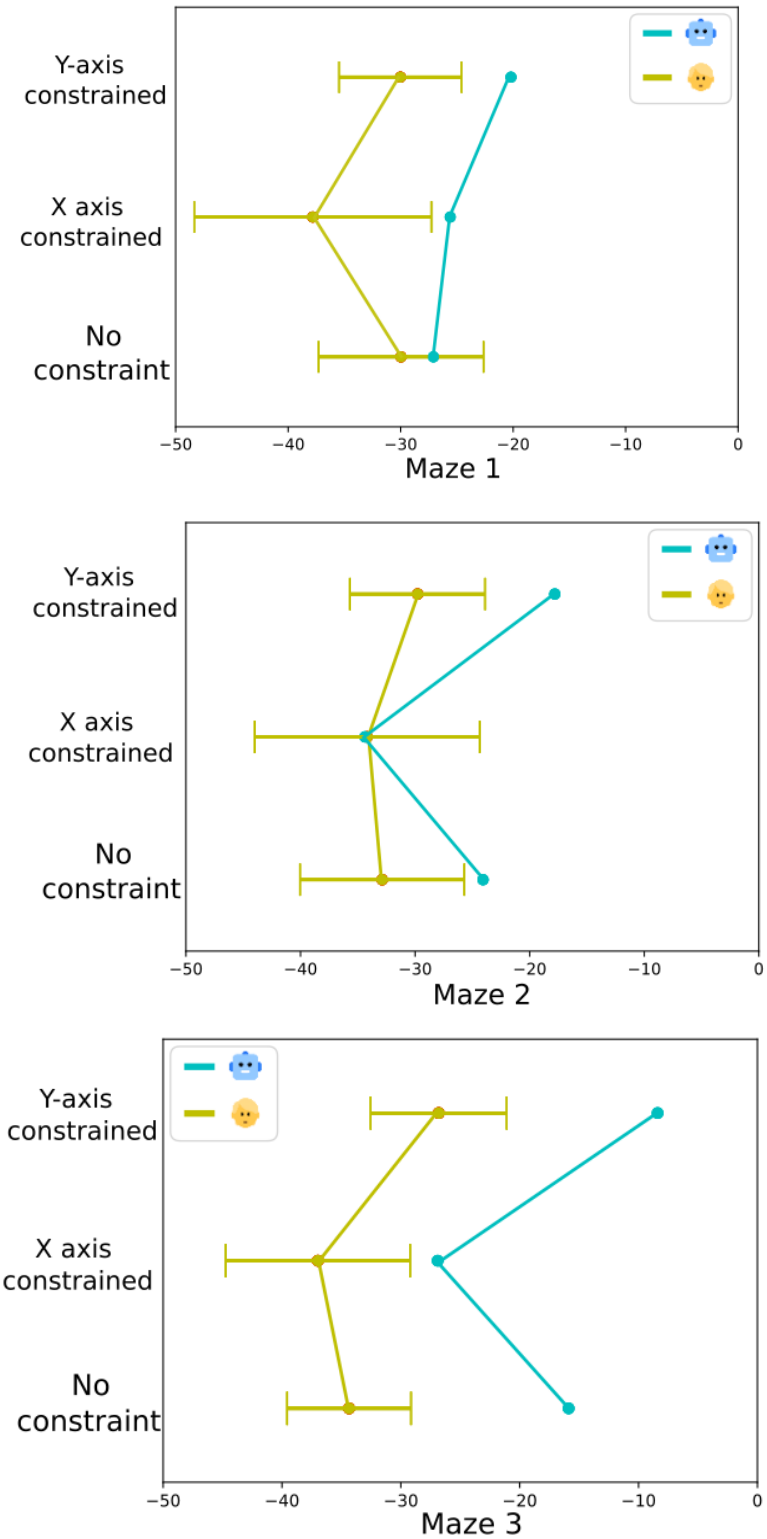


FIGURE 6.24 – *Front-Back Speed : Expert Teacher versus Agent*
In this figure we compare Expert Teacher (in yellow) Front-Back axis average speed versus the trained Agent one (in cyan) across our 3 conditions. We notice that the speed in the Front-Back constrained condition is significantly decreased on both Agent and Human Expert compared to the Unconstrained condition.
We also notice in 2 cases out of 3 that the Right-Left constrained condition's speed is significantly higher on both Human Expert and the Agent compared to the Unconstrained condition.

6.14 Future work : Haptic Guidance

Haptic guidance in virtual or teleoperation environment have, for a long time, used virtual fixtures [Rosenberg, 1993] or artificial force fields [Xiao and Hubbold, 1998] in order to create control assistance and help the user move toward goal or prevent the user to enter some areas. Robotics improved the uses of haptic devices due to the simplicity to create haptic feedback (most robots have force or distance sensors) and the necessity to operate non autonomous robot more accurately with improved stability [Bettini et al., 2004]. Implementation wise haptic feedback greatly widened its range during the last decades including haptic control devices, wrist haptic devices [Goto et al., 2018], gloves, suits, steering wheels for application like parking [Tada et al., 2016], for fatigue-related behavior enhancement, for teleoperated car control [Hosseini et al., 2016], for teaching curve to new drivers [Mulder et al., 2008], teaching AirHockey [Moon and Seo, 2021, 2019] It could improve fields like Computer interfaces, Virtual Reality, driving assistance, robot teleoperation, UAV (unmanned aerial vehicle) like aerial and marine/submarine, drones, surgical robotics and the teaching of these fields to novice users.

6.15 Conclusion

Using a Deep Learning agent trained on our Human Expert Data, our agent learnt multiple aspects of the task and of the human expert behavior. Our agent learnt visual features like collision avoidance, color related direction, human trajectory behaviors and human velocity behaviors. We compared our agent motion and velocities to the human participants to prove that our agent has human like behavior. Our agent was taught only from our human expert so it learnt only from him. But our Human Expert was himself so trained with the haptic feedback device that some particularity appeared between our participant velocity and our human expert's one. This showed our agent really learned specific behaviors in only particular cases. This led to our Agent being closer to what we had hoped of our Human Expert.

CHAPTER 7

DISCUSSION AND CONCLUSION

Contents

7.1 Achievements

In this thesis we explored embodied robot control, human impedance estimation and learning from demonstration.

We first achieved an intuitive admittance control on a wearable robot arm based on force sensor levers that control each servo-motor controlled based arms. We introduce a simple user interface for supernumerary wearable robot arm systems. This interface utilizes a force sensor to enable users to guide the robot arms and enable various passive and active assistance tasks, and enabling the user to teach simple movements to the robot. The key feature of this user interface is its intuitiveness and ease of use. We currently demonstrated the versatility of the interface in three modes of application, and we are now developing its applications in the field of elderly care where we believe it can be extremely useful.

Then we brought embodiment to robotics by implementing our own embodied robot system with a Haption Virtuose 3D haptic feedback device, a Franka Panda 7DoF robot arm and a VR head mounted display coupled to a 360° camera. We exploited impedance control and quadratic programming to operate safely the robot teleoperation.

We used this embodied robot setup to implement a novel human arm impedance estimation method based on task-induced vibrations and applied this estimation to the impedance controlled robot, achieving tele-impedance. For this purpose, we propose a methodology for online impedance estimation, and online reference estimation from the human operator. We designed the controller envisaging use in assistive scenarios, where a human physiotherapist or caregiver guides a follower robot to help elderly patients. These scenarios are characterized by perturbations from the individual and hence we propose to estimate the human impedance directly from the perturbations.

For this purpose, we propose a methodology for online impedance estimation, and online reference estimation from the human operator.

As mentioned in the experiments, we were unable to use the damping calculated from the human operator on our robot. We found these values too low, relative to the stiffness, to ensure stable robot behavior. This comes as no surprise, given the different inertia of human arm and robot. Ideally our controller, in which the robot forces are fed to the operator and the operator movements are sent to the robot, should impose the human arm dynamics on the robot, therefore theoretically ensuring that the human damping ratios are sufficient for the robot. In practice though, this is possible only if the mass of the robot is well compensated for, which is not a trivial challenge.

Human interactive behaviors are enabled by simultaneous adaptations of force, trajectory and impedance. These adaptation are both predictive, to ensure stability when an interaction starts, as well as reactive, to maintain the stability during perturbations in a task. The method we propose here is specific for the measurement of reactive impedance and is arguably better than muscle activation (EMG or grip force) based impedance estimations in the presence of perturbations (as discussed in the introduction). On the other hand, muscle activation based techniques are the only ones available for impedance estimations before the start of a movement, and in the absence of sufficient external perturbations. Robust impedance estimation in real world tasks therefore requires us to develop an integrated estimation framework in which the predictive impedance changes can be measured using muscle activation (via EMG or grip force) and the reactive changes are estimated using the perturbations, like we propose here in this study.

Lastly we used this same embodied robot setup to bring embodiment and Learning from Demonstration together. By using a Deep Learning agent trained on our Human Expert Data, we could have our agent learn multiple aspect of our task and of the human expert behavior. We used state-of-the-art visual recognition network ResNET, Gated Recurrent Units and Mixtures Density Networks to train from short video clip to get our Agent have planar motion. Our agent learnt visual feature like collision avoidance, color related direction, human position behaviors and human velocity trajectories without us never having explicitly write any policy or explain the constraints or dynamics. We compared our agent motion and velocities to our human participants to prove that our agent has human like behavior. This was proved for most of the behaviors, but unfortunately our agent learnt only from our Human Expert that was himself so trained with the haptic feedback device that some particularity appeared between our participant velocity and our human expert's one. But this showed our agent could learn behaviors specific to our Human Expert.

7.2 Future work

Our agent learnt well some human particularities in simulation but we never had the opportunity to bring it in a real life environment. In the future we would like to bring it to real life by applying transfer learning with a real life dataset.

We also noticed our Agent had learnt specific behaviors of our Human Expert that could not be found in the novice participants behaviors. This means that by creating the dataset whith our system, our Expert acquired experience that was not acquired by our novice participants. Hence it might become difficult to learn a specific behavior of a unique novice participant since its behavior changes as long as he/she trains our Agent. A possible solution could be to decrease the dataset length, but it could have negative effect on the training, others would be to apply some transfer learning to our current Agent with a shorter dataset created by a novice participant.

Also we think a new field can be studied with this agent and this way of training. The field of haptic guidance, mainly in order to guide medical robots, could gain by having these methods applied to. Indeed our agent showed it was able to catch individual details of our Human Expert that our human participant panel did not possessed. The learning process being completely intuitive for the human expert, inputs are taken from images and labels from the guided robot position. This could lead to learn expert way to have guidance through the haptic system, such as teleoperated medical robots, to teach new users move as an expert and understand their wrong moves.

CHAPTER 8

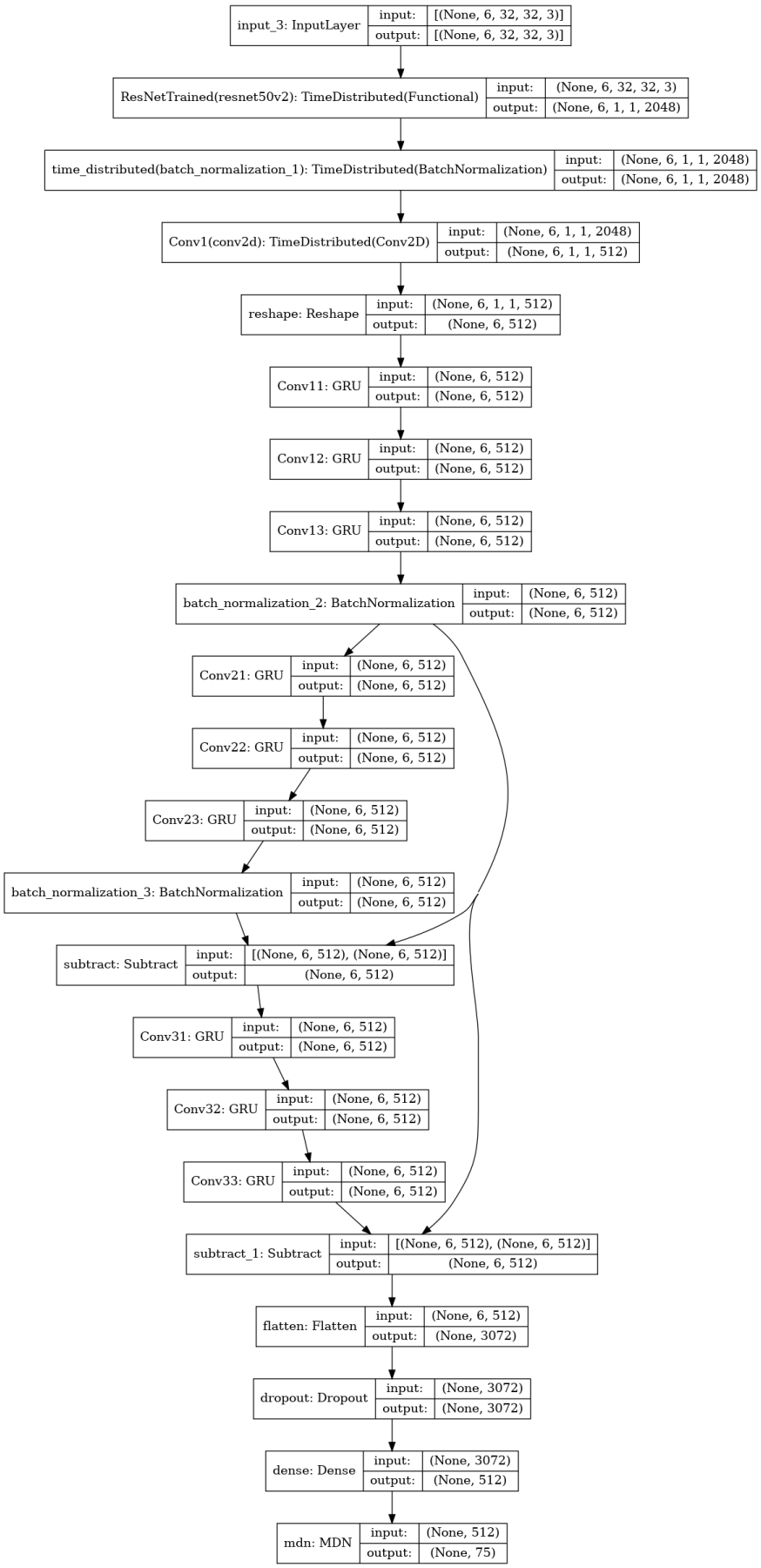
APPENDIX

Appendix A

Example of non sequential model description in TensorFlow

```
def GRU_MDN_model(IMG_SIZE,WINDOW_SIZE):

    ResNet = ResNet50V2(
    include_top= None, weights='imagenet', input_tensor=None,
       pooling=None, input_shape=(IMG_SIZE, IMG_SIZE,3))
    ResNet.trainable = True

    model = tf.keras.Sequential()
    dr = 0.3 #Dropout rate
    unit =512 #Units number

    init_lr = 0.0001 # Learning rate
    decay_steps = 1000

    IMG = layers.Input(shape=(WINDOW_SIZE,IMG_SIZE,IMG_SIZE,3))

    RNfeature =
       layers.TimeDistributed(ResNet,name="ResNetTrained")(IMG)
    norm = layers.BatchNormalization()(IMG)

    conv =
       layers.TimeDistributed(layers.BatchNormalization())(RNfeature)

    conv = layers.TimeDistributed(layers.Conv2D(kernel_size=2,
       filters=unit, activation='relu',
       padding="same"),name="Conv1")(conv)

    conv = layers.Reshape((WINDOW_SIZE,unit))(conv)

    gru = layers.GRU(unit, return_sequences=True, dropout=dr)(conv)
    gru = layers.GRU(unit, return_sequences=True, dropout=dr)(gru)
    gru = layers.GRU(unit, return_sequences=True, dropout=dr)(gru)
```

```
mid1 = layers.BatchNormalization()(gru)

gru = layers.GRU(unit, return_sequences=True, dropout=dr)(mid1)
gru = layers.GRU(unit, return_sequences=True, dropout=dr)(gru)
gru = layers.GRU(unit, return_sequences=True, dropout=dr)(gru)
mid2 = layers.BatchNormalization()(conv1)
mid12 = layers.Subtract()([mid2, mid1])

gru = layers.GRU(unit, return_sequences=True, dropout=dr)(mid12)
gru = layers.GRU(unit, return_sequences=True, dropout=dr)(gru)
end3 = layers.GRU(unit, return_sequences=True, dropout=dr)(gru)

end3 = layers.Subtract()([end3, mid1])

outlayer = layers.Flatten()(end3)
outlayer = Dropout(dr)(outputs)

padding="same",name="Conv2D4"))(end3)
N_HIDDEN=unit #Hidden Unit number of the MDN
OUTPUT_DIMS=2 #Output dimension for the MDN
N_MIXES=15 #Number of mixing coefficient for the MDN


lr_decayed_fn =
   tf.keras.optimizers.schedules.CosineDecay(init_lr, decay_steps)


outlayer = layers.Dense(N_HIDDEN, batch_input_shape=(None, 1),
   activation='relu')(outlayer)

outlayer = mdn.MDN(OUTPUT_DIMS, N_MIXES)(outputs)

model = tf.keras.Model(inputs=[IMG], outputs=[outlayer])
model.compile(loss=mdn.get_mixture_loss_func(OUTPUT_DIMS,N_MIXES),optimizer=o
   =init_lr))

return model
```



– ***Appendix B*** *: Our architecture containing TensorFlow name and shape of every layer*

BIBLIOGRAPHIE

18

Openai,2018. URL https://openai.com/blog/learning-dexterity/. 56

United nations report (2010). world aging population 2009 (publication no. st/-sea/ser.a/295). URL http://www.un.org/esa/population/publications/WPA2009/WPA2009-report.pdf. 37

METI Report 2014. Meti 2014 revision of the four priority areas to which robot technology is to be introduced in nursing care of the elderly. URL http://www.meti.go.jp/english/press/2014/0203_02.html. 38

Worldbank Report 2016. World bank report (2016). URL http://data.worldbank.org/indicator/SP.POP.65UP.TO.ZS. 37

A. Ajoudani, N. G. Tsagarakis, and A. Bicchi. Tele-impedance: Towards transferring human impedance regulation skills to robots. In *2012 IEEE International Conference on Robotics and Automation*. IEEE, May 2012. doi: 10.1109/icra.2012.6224904. URL https://doi.org/10.1109/icra.2012.6224904. 9, 38

Arash Ajoudani. *Transferring Human Impedance Regulation Skills to Robots*. Springer International Publishing, 2016. doi: 10.1007/978-3-319-24205-7. URL https://doi.org/10.1007/978-3-319-24205-7. 9, 10, 39, 40

Fernando Almeida, António Lopes, and Paulo Abreu. Force-impedance control: a new control strategy of robotic manipulators. *Recent advances in Mechatronics*, 1:126–137, 1999. 7

Luis Almeida, Bruno Patrao, Paulo Menezes, and Jorge Dias. Be the robot: Human embodiment in tele-operation driving tasks. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, August 2014. doi: 10.1109/roman.2014.6926298. URL https://doi.org/10.1109/roman.2014.6926298. 11

Jumpei Arata, Masashi Hattori, Shohei Ichikawa, and Masamichi Sakaguchi. Robotically enhanced rubber hand illusion. *IEEE Transactions on Haptics*, 7(4):526–532, 2014. doi: 10.1109/TOH.2014.2304722. 11

Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57 (5):469–483, 2009. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2008.10.024. URL https://www.sciencedirect.com/science/article/pii/S0921889008001772. 16

Laura Aymerich-Franch, Damien Petit, Gowrishankar Ganesh, and Abderrahmane Kheddar. The second me: Seeing the real body during humanoid robot embodiment produces an illusion of bi-location. *Consciousness and Cognition*, 46:99–109, 2016. ISSN 1053-8100. doi: https://doi.org/10.1016/j.concog.2016.09.017. URL https://www.sciencedirect.com/science/article/pii/S1053810016303038. 11

Laura Aymerich-Franch, Damien Petit, Gowrishankar Ganesh, and Abderrahmane Kheddar. Object Touch by a Humanoid Robot Avatar Induces Haptic Sensation in the Real Hand. *Journal of Computer-Mediated Communication*, 22(4):215–230, 07 2017. ISSN 1083-6101. doi: 10.1111/jcc4.12188. URL https://doi.org/10.1111/jcc4.12188. 11

J.A. Bagnell and J.G. Schneider. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, volume 2, pages 1615–1620 vol.2, 2001. doi: 10.1109/ROBOT.2001.932842. 15

A. Bettini, P. Marayong, S. Lang, A.M. Okamura, and G.D. Hager. Vision-assisted control for manipulation using virtual fixtures. *IEEE Transactions on Robotics*, 20(6): 953–966, December 2004. doi: 10.1109/tro.2004.829483. URL https://doi.org/10.1109/tro.2004.829483. 88

Christopher M. Bishop. Mixture density networks, 1994. 67

A. Bolopion and S. Regnier. A review of haptic feedback teleoperation systems for micromanipulation and microassembly. *IEEE Transactions on Automation Science and Engineering*, 10(3):496–502, July 2013. doi: 10.1109/tase.2013.2245122. URL https://doi.org/10.1109/tase.2013.2245122. 6

Matthew Botvinick and Jonathan Cohen. Rubber hands 'feel' touch that eyes see. *Nature*, 391(6669):756–756, February 1998. doi: 10.1038/35784. URL https://doi.org/10.1038/35784. 11

Karim Bouyarmane, Joris Vaillant, Kévin Chappellet, and Abderrahmane Kheddar. Multi-robot and task-space force control with quadratic programming. working paper or preprint, March 2017. URL https://hal.archives-ouvertes.fr/hal-01495662. 35

J. Broekens, M. Heerink, and H. Rosendal. Assistive social robots in elderly care: a review. *Gerontechnology*, 8(2), April 2009. doi: 10.4017/gt.2009.08.02.002.00. URL https://doi.org/10.4017/gt.2009.08.02.002.00. 38

Etienne Burdet, Rieko Osu, David W. Franklin, Theodore E. Milner, and Mitsuo Kawato. The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature*, 414(6862):446–449, November 2001. doi: 10.1038/35106566. URL https://doi.org/10.1038/35106566. 9

Sylvain Calinon. *Learning from Demonstration (Programming by Demonstration)*, pages 1–8. Springer Berlin Heidelberg, Berlin, Heidelberg, 2018. ISBN 978-3-642-41610-1. doi: 10.1007/978-3-642-41610-1_27-1. URL https://doi.org/10.1007/978-3-642-41610-1_27-1. 16

A. Canziani, Adam Paszke, and Eugenio Culurciello. An analysis of deep neural network models for practical applications. *ArXiv*, abs/1605.07678, 2016. 55

Zuyan Chen, Jared Walters, Gang Xiao, and Shuai Li. An enhanced gru model with application to manipulator trajectory tracking. *EAI Endorsed Transactions on AI and Robotics*, 1:1–11, Jan. 2022. doi: 10.4108/airo.v1i.7. URL https://publications.eai.eu/index.php/airo/article/view/7. 67

Alexandre Cherpillod, Stefano Mintchev, and Dario Floreano. Embodied flight with a drone. *CoRR*, abs/1707.01788, 2017. URL http://arxiv.org/abs/1707.01788. 12

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014. URL https://arxiv.org/abs/1409.1259. 56

Sungjoon Choi, Kyungjae Lee, Sungbin Lim, and Songhwai Oh. Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6915–6922, 2018. doi: 10.1109/ICRA.2018.8462978. 68

Dan Cireşan, Ueli Meier, and Juergen Schmidhuber. Multi-column deep neural networks for image classification. 2012. doi: 10.48550/ARXIV.1202.2745. URL https://arxiv.org/abs/1202.2745. 66

R. Cortesao, Jaeheung Park, and O. Khatib. Real-time adaptive control for haptic telemanipulation with kalman active observers. *IEEE Transactions on Robotics*, 22(5): 987–999, 2006. doi: 10.1109/TRO.2006.878787. 38

Kerstin Dautenhahn and Chrystopher L. Nehaniv. *Imitation in Animals and Artifacts (Complex Adaptive Systems)*. A Bradford Book, hardcover edition, 5 2002. ISBN 978-0397510795. URL https://lead.to/amazon/com/?op=bt&la=en&cu=usd&key=0262042037. 16

Marc Deisenroth and Carl Rasmussen. Pilco: A model-based and data-efficient approach to policy search. pages 465–472, 01 2011. 15

Luuk Maria Doornebosch, David A. Abbink, and Luka Peternel. Analysis of coupling effect in human-commanded stiffness during bilateral tele-impedance. *IEEE Transactions on Robotics*, pages 1–16, 2021. doi: 10.1109/tro.2020.3047064. URL https://doi.org/10.1109/tro.2020.3047064. 9, 10, 39

Theodore E. Milner Etienne Burdet, David W. Franklin. *Human Robotics*, volume 978-0-262-01953-8. MIT Press, 2013. URL https://mitpress.mit.edu/books/human-robotics. 41, 47

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400, 2017. URL http://arxiv.org/abs/1703.03400. 15

Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards, 2015. URL https://arxiv.org/abs/1511.07404. 18

David W. Franklin, Gary Liaw, Theodore E. Milner, Rieko Osu, Etienne Burdet, and Mitsuo Kawato. Endpoint stiffness of the arm is directionally tuned to instability in the environment. *Journal of Neuroscience*, 27(29):7705–7716, 2007. ISSN 0270-6474. doi: 10.1523/JNEUROSCI.0968-07.2007. URL https://www.jneurosci.org/content/27/29/7705. 41, 47

G. Ganesh, A. Takagi, R. Osu, T. Yoshioka, M. Kawato, and E. Burdet. Two is better than one: Physical interactions improve motor performance in humans. *Scientific Reports*, 4(1), January 2014. doi: 10.1038/srep03824. URL https://doi.org/10.1038/srep03824. 21, 38

Gowrishankar Ganesh and Etienne Burdet. Motor planning explains human behaviour in tasks with multiple solutions. *Robotics and Autonomous Systems*, 61(4):362–368, April 2013. doi: 10.1016/j.robot.2012.09.024. URL https://doi.org/10.1016/j.robot.2012.09.024. 41

Gowrishankar Ganesh, Alin Albu-Schaffer, Masahiko Haruno, Mitsuo Kawato, and Etienne Burdet. Biomimetic motor behavior for simultaneous adaptation of force, impedance and trajectory in interaction tasks. In *2010 IEEE International Conference on Robotics and Automation*. IEEE, May 2010. doi: 10.1109/robot.2010.5509994. URL https://doi.org/10.1109/robot.2010.5509994. 9, 38

E. Burdet G.Ganesh, A.Melendez-Calderon M.Kawato. Transition between reciprocal activation and co-contraction during wrist posture control. 2020. URL https://hal.archives-ouvertes.fr/hal-02974058. 41

H. Gomi and R. Osu. Contributions of single and double joint stiffness of human arm during force control. In *Proceedings of 18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 5, pages 2244–2245 vol.5, 1996. doi: 10.1109/IEMBS.1996.646516. 9

Hiroaki Gomi and Mitsuo Kawato. Human arm stiffness and equilibrium-point trajectory during multi-joint movement. *Biological Cybernetics*, 76(3):163–171, April 1997. doi: 10.1007/s004220050329. URL https://doi.org/10.1007/s004220050329. 38

Hiroaki Gomi and Rieko Osu. Task-dependent viscoelasticity of human multijoint arm and its spatial characteristics for interaction with environments. *The Journal of Neuroscience*, 18(21):8965–8978, November 1998. doi: 10.1523/jneurosci.18-21-08965.1998. URL https://doi.org/10.1523/jneurosci.18-21-08965.1998. 9

Takashi Goto, Swagata Das, Yuichi Kurita, and Kai Kunze. Artificial motion guidance. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings*. ACM, October 2018. doi: 10.1145/3266037.3271644. URL https://doi.org/10.1145/3266037.3271644. 88

Arvid Guterstam, Valeria I Petkova, and H Henrik Ehrsson. The illusion of owning a third arm. *PLoS One*, 6(2):e17208, February 2011. 11

Mathias Haage, Grigoris Piperagkas, Christos Papadopoulos, Ioannis Mariolis, Jacek Malec, Yasemin Bekiroglu, Mikael Hedelind, and Dimitrios Tzovaras. Teaching assembly by demonstration using advanced human robot interaction and a knowledge

integration framework. *Procedia Manufacturing*, 11:164–173, 2017. doi: 10.1016/j.promfg.2017.07.221. URL https://doi.org/10.1016/j.promfg.2017.07.221. 24

Masayuki Hara, Hiroyuki Nabae, Akio Yamamoto, and Toshiro Higuchi. A novel rubber hand illusion paradigm allowing active self-touch with variable force feedback controlled by a haptic device. *IEEE Transactions on Human-Machine Systems*, 46(1): 78–87, 2016. doi: 10.1109/THMS.2015.2487499. 11

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90. 66

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1044. URL https://aclanthology.org/P17-1044. 73

Matthew D. Hill and Gunter Niemeyer. Real-time estimation of human impedance for haptic interfaces. In *World Haptics 2009 - Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. IEEE, 2009. doi: 10.1109/whc.2009.4810893. URL https://doi.org/10.1109/whc.2009.4810893. 9, 39, 40

Koh Hiraoka, Seiichiro Tateishi, and Koji Mori. Review of health issues of workers engaged in operations related to the accident at the fukushima daiichi nuclear power plant. *J. Occup. Health*, 57(6):497–512, 2015. 12

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997. doi: 10.1162/neco.1997.9.8.1735. 56

Neville Hogan. Impedance control: An approach to manipulation. In *1984 American Control Conference*, pages 304–313, 1984. doi: 10.23919/ACC.1984.4788393. 7

Arata Horie, Hideki Shimobayashi, and Masahiko Inami. Torsioncrowds: Multi-points twist stimulation display for large part of the body. In *ACM SIGGRAPH 2020 Emerging Technologies*, SIGGRAPH '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379670. doi: 10.1145/3388534.3407303. URL https://doi.org/10.1145/3388534.3407303. 8

Amin Hosseini, Florian Richthammer, and Markus Lienkamp. Predictive haptic feedback for safe lateral control of teleoperated road vehicles in urban areas. In *2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)*, pages 1–7, 2016. doi: 10.1109/VTCSpring.2016.7504430. 88

https://www.franka.de/technology. Franka panda emika 7dof torque controlled robot arm. 30, 33

https://www.haption.com/fr/products-fr/virtuose-3d fr.html. Haption virtuose 3d, haptic feedback device. 30, 32

https://www.vive.com/fr/product/vive pro/. Vr head mounted display htc vive. 30, 33

Ronny Hug, Stefan Becker, Wolfgang Hübner, and Michael Arens. Particle-based pedestrian path prediction using lstm-mdl models. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2684–2691, 2018. doi: 10.1109/ITSC.2018.8569478. 56

T. Imaida, Y. Yokokohji, T. Doi, M. Oda, and T. Yoshikawa. Ground-space bilateral teleoperation of ets-vii robot arm by direct bilateral coupling under 7-s time delay condition. *IEEE Transactions on Robotics and Automation*, 20(3):499–511, 2004. doi: 10.1109/TRA.2004.825271. 6

Yukiko Iwasaki, Benjamin Navarro, Hiroyasu Iwata, and Gowrishankar Ganesh. Embodiment modifies attention allotment for the benefit of dual task performance. *Commun. Biol.*, 5(1):701, July 2022. 11, 12, 13, 62

Atsushi Izumihara, Tomoya Sasaki, Masahiro Ogino, Reona Takamura, and Masahiko Inami. Transfantome: Transformation into bodies of various scale and structure in multiple spaces. In *ACM SIGGRAPH 2019 Emerging Technologies*, SIGGRAPH '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450363082. doi: 10.1145/3305367.3327980. URL https://doi.org/10.1145/3305367.3327980. 21

Seul Jung, Tien C Hsia, and Robert G Bonitz. Force tracking impedance control of robot manipulators under unknown environment. *IEEE Transactions on Control Systems Technology*, 12(3):474–483, 2004. 7

Sunggoo Jung, Sunyou Hwang, Heemin Shin, and David Hyunchul Shim. Perception, guidance, and navigation for indoor autonomous drone racing using deep learning. *IEEE Robotics and Automation Letters*, 3(3):2539–2544, July 2018. doi: 10.1109/lra.2018.2808368. URL https://doi.org/10.1109/lra.2018.2808368. 18

Saki Kato, Natsuki Yamanobe, Gentiane Venture, Eiichi Yoshida, and Gowrishankar Ganesh. The where of handovers by humans: Effect of partner characteristics, distance and visual feedback. *PLOS ONE*, 14(6):e0217129, June 2019. doi: 10.1371/journal.pone.0217129. URL https://doi.org/10.1371/journal.pone.0217129. 38

Parham M Kebria, Hamid Abdi, Mohsen Moradi Dalvand, Abbas Khosravi, and Saeid Nahavandi. Control methods for internet-based teleoperation systems: A review. *IEEE Transactions on Human-Machine Systems*, 49(1):32–46, 2018. 6

Oussama Khatib, Xiyang Yeh, Gerald Brantner, Brian Soe, Boyeon Kim, Shameek Ganguly, Hannah Stuart, Shiquan Wang, Mark Cutkosky, Aaron Edsinger, Phillip Mullins, Mitchell Barham, Christian R. Voolstra, Khaled Nabil Salama, Michel L'Hour, and Vincent Creuze. Ocean one: A robotic avatar for oceanic discovery. *IEEE Robotics Automation Magazine*, 23(4):20–29, 2016. doi: 10.1109/MRA.2016.2613281. 1, 12

Konstantina Kilteni, Raphaela Groten, and Mel Slater. The Sense of Embodiment in Virtual Reality. *Presence: Teleoperators and Virtual Environments*, 21(4):373–387, 11 2012. doi: 10.1162/PRES_a_00124. URL https://doi.org/10.1162/PRES_a_00124. 11

Chung Min Kim, Michael Danielczuk, Isabella Huang, and Ken Goldberg. Ipc-graspsim: Reducing the sim2real gap for parallel-jaw grasping with the incremental potential contact model. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6180–6187, 2022. doi: 10.1109/ICRA46639.2022.9811777. 18

Tobias Klamt, Diego Rodriguez, Lorenzo Baccelliere, Xi Chen, Domenico Chiaradia, Torben Cichon, Massimiliano Gabardi, Paolo Guria, Karl Holmquist, Malgorzata Kamedula, Hakan Karaoguz, Navvab Kashiri, Arturo Laurenzi, Christian Lenz, Daniele Leonardis, Enrico Mingo Hoffman, Luca Muratore, Dmytro Pavlichenko, Francesco Porcini, Zeyu Ren, Fabian Schilling, Max Schwarz, Massimiliano Solazzi, Michael Felsberg, Antonio Frisoli, Michael Gustmann, Patric Jensfelt, Klas Nordberg, Jürgen Roßmann, Uwe Süss, Nikos G. Tsagarakis, and Sven Behnke. Flexible disaster response of tomorrow - final presentation and evaluation of the CENTAURO system. *CoRR*, abs/1909.08812, 2019. URL http://arxiv.org/abs/1909.08812. 12

J. Zico Kolter and Andrew Y Ng. The stanford littledog: A learning and rapid replanning approach to quadruped locomotion. *The International Journal of Robotics Research*, 30(2):150–174, 2011. doi: 10.1177/0278364910390537. URL https://doi.org/10.1177/0278364910390537. 15

Fukushima Kunihiko. 62(10):p658–665, 10 1979. ISSN 03736091. URL https://cir.nii.ac.jp/crid/1522262180163676160. 54

Pablo Lanillos. Robot self other distinction active inference meets neural networks learning in a mirror, 2020. URL https://underline.io/lecture/1800-robot-self-other-distinction-active-inference-meets-neural-network 68

Heedon Lee, Wansoo Kim, Jungsoo Han, and Changsoo Han. The technical trend of the exoskeleton robot system for human power assistance. *International Journal of Precision Engineering and Manufacturing*, 13(8):1491–1497, Aug 2012. ISSN 2005-4602. doi: 10.1007/s12541-012-0197-x. URL https://doi.org/10.1007/s12541-012-0197-x. 24

Paulo Leica, Karen Rivera, Stalin Muela, Danilo Chavez, Gabriela Andaluz, and Victor H. Andaluz. Consensus algorithms for bidirectional teleoperation of aerial manipulator robots in an environment with obstacles. In *2019 IEEE Fourth Ecuador Technical Chapters Meeting (ETCM)*. IEEE, November 2019. doi: 10.1109/etcm48019.2019.9014872. URL https://doi.org/10.1109/etcm48019.2019.9014872. 6

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies, 2015. URL https://arxiv.org/abs/1504.00702. 18

Y. Li, G. Ganesh, N. Jarrassé, S. Haddadin, A. Albu-Schaeffer, and E. Burdet. Force, impedance, and trajectory learning for contact tooling and haptic identification. *IEEE Transactions on Robotics*, 34(5):1170–1182, 2018. doi: 10.1109/TRO.2018.2830405. 38

Yugang Liu and Goldie Nejat. Robotic urban search and rescue: A survey from the control perspective. *Journal of Intelligent &amp Robotic Systems*, 72(2):147–165, March 2013. doi: 10.1007/s10846-013-9822-x. URL https://doi.org/10.1007/s10846-013-9822-x. 1

B. Llorens-Bonilla, F. Parietti, and H. Asada. Demonstration-based control of supernumerary robotic limbs, ieee international conference on intelligent robots and systems. *IROS*, 2012. 19

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2016. URL https://arxiv.org/abs/1608.03983. 59

Manolis Loukadakis, José Cano, and Michael O'Boyle. Accelerating deep neural networks on low power heterogeneous architectures. 01 2018. 55

Han Luo, Mingzhu Wang, Peter Kok-Yiu Wong, Jingyuan Tang, and Jack C.P. Cheng. Construction machine pose prediction considering historical motions and activity attributes using gated recurrent unit (GRU). *Automation in Construction*, 121:103444, January 2021. doi: 10.1016/j.autcon.2020.103444. URL https://doi.org/10.1016/j.autcon.2020.103444. 64

Jing Luo, Chenguang Yang, Ning Wang, and Min Wang. Enhanced teleoperation performance using hybrid control and virtual fixture. *International Journal of Systems Science*, 50(3):451–462, January 2019. doi: 10.1080/00207721.2018.1562128. URL https://doi.org/10.1080/00207721.2018.1562128. 9, 38, 40

Azumi Maekawa, Shota Takahashi, MHD Yamen Saraiji, Sohei Wakisaka, Hiroyasu Iwata, and Masahiko Inami. Naviarm. In *Proceedings of the 10th Augmented Human International Conference 2019*. ACM, March 2019. doi: 10.1145/3311823.3311849. URL https://doi.org/10.1145/3311823.3311849. 21

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance gpu-based physics simulation for robot learning, 2021. 15

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602. 18

Hee-Seung Moon and Jiwon Seo. Prediction of human trajectory following a haptic robotic guide using recurrent neural networks. In *2019 IEEE World Haptics Conference (WHC)*. IEEE, July 2019. doi: 10.1109/whc.2019.8816157. URL https://doi.org/10.1109/whc.2019.8816157. 88

Hee-Seung Moon and Jiwon Seo. Optimal action-based or user prediction-based haptic guidance: Can you do even better? *CoRR*, abs/2101.01870, 2021. URL https://arxiv.org/abs/2101.01870. 63, 88

Roozbeh Mottaghi, Hessam Bagherinezhad, Mohammad Rastegari, and Ali Farhadi. Newtonian image understanding: Unfolding the dynamics of objects in static images, 2015. URL https://arxiv.org/abs/1511.04048. 18

T. Mouri, H. Kawasaki, and S. Ueki. Bilateral tele-operated hand robot with communicational time delay. *IFAC-PapersOnLine*, 50(1):12721–12726, 2017. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2017.08.1824. URL https://www.sciencedirect.com/science/article/pii/S2405896317324461. 20th IFAC World Congress. 38

Mark Mulder, David A. Abbink, and Erwin R. Boer. The effect of haptic guidance on curve negotiation behavior of young, experienced drivers. In *2008 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, October 2008. doi: 10.1109/icsmc.2008.4811377. URL https://doi.org/10.1109/icsmc.2008.4811377. 88

N. Muramatsu and H. Akiyama. Japan: Super-aging society preparing for the future. *The Gerontologist*, 51(4):425–432, July 2011. doi: 10.1093/geront/gnr067. URL https://doi.org/10.1093/geront/gnr067. 37

Koki NAKABAYASHI, Yukiko Iwasaki, Shota Takahashi, and Hiroyasu IWATA. Research on "third arm": voluntarily operative wearable robot arm: - design of wearable robot arm having high working properties and low workspace invasiveness -. *The Proceedings of JSME annual Conference on Robotics and Mechatronics (Robomec)*, 2017:1P1–M09, 11 2017. doi: 10.1299/jsmermd.2017.1P1-M09. 19

Yashraj Narang, Kier Storey, Iretiayo Akinola, Miles Macklin, Philipp Reist, Lukasz Wawrzyniak, Yunrong Guo, Adam Moravanszky, Gavriel State, Michelle Lu, Ankur Handa, and Dieter Fox. Factory: Fast contact for robotic assembly, 2022. URL https://arxiv.org/abs/2205.03532. 15, 18

Benjamin Navarro, Aïcha Fonte, Philippe Fraisse, Gérard Poisson, and Andrea Cherubini. Introducing OpenPHRI: a software library for physical human-robot interaction. In *ICRA 2018 Late Breaking Posters*, Brisbane, Australia, May 2018. IEEE. URL https://hal.archives-ouvertes.fr/hal-01734741. Late breaking result abstract. 22

Alex Nichol, Vicki Pfau, Christopher Hesse, Oleg Klimov, and John Schulman. Gotta learn fast: A new benchmark for generalization in rl. 2018. doi: 10.48550/ARXIV. 1804.03720. URL https://arxiv.org/abs/1804.03720. 17

Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, and Dongjun Lee. Telerobotics. In *Springer Handbook of Robotics*, pages 1085–1108. Springer International Publishing, 2016. doi: 10.1007/978-3-319-32552-1_43. URL https://doi.org/10.1007/978-3-319-32552-1_43. 38

OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving rubik's cube with a robot hand, 2019. 15

Ana-Lucia Pais Ureche and Aude Billard. Learning bimanual coordinated tasks from human demonstrations. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*, HRI'15 Extended Abstracts, page 141–142, New York, NY, USA, 2015. Association for Computing Machinery. ISBN 9781450333184. doi: 10.1145/2701973.2702007. URL https://doi.org/10.1145/2701973.2702007. 16

Evangelos Papadopoulos, Farhad Aghili, Ou Ma, and Roberto Lampariello. Robotic manipulation and capture in space: A survey. *Front. Robot. AI*, 8:686723, 2021. 1

Raj Patel, Meysar Zeinali, and Kalpdrum Passi. Deep learning-based robot control using recurrent neural networks (lstm; gru) and adaptive sliding mode control. 67

Michael A Peshkin, J Edward Colgate, Wit Wannasuphoprasit, Carl A Moore, R Brent Gillespie, and Prasad Akella. Cobot architecture. *IEEE Transactions on Robotics and Automation*, 17(4):377–390, 2001. 20

Luka Peternel, Tadej Petrič, and Jan Babič. Robotic assembly solution by human-in-the-loop teaching method based on real-time stiffness modulation. *Autonomous Robots*, 42(1):1–17, April 2017. doi: 10.1007/s10514-017-9635-z. URL https://doi.org/10.1007/s10514-017-9635-z. 9, 38

Susanne Poeller, Max V. Birk, Nicola Baumann, and Regan L. Mandryk. Let me be implicit: Using motive disposition theory to predict and explain behaviour in digital games. CHI '18, page 1–15, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356206. doi: 10.1145/3173574.3173764. URL https://doi.org/10.1145/3173574.3173764. 63

Harish Ravichandar, Athanasios S Polydoros, Sonia Chernova, and Aude Billard. Recent advances in robot learning from demonstration. *Annu. Rev. Control Robot. Auton. Syst.*, 3(1):297–330, 2020. 15, 16, 17

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. doi: 10.1109/CVPR.2016.91. 57

Nils Reimers and Iryna Gurevych. Optimal hyperparameters for deep lstm-networks for sequence labeling tasks. *CoRR*, abs/1707.06799, 2017. URL http://arxiv.org/abs/1707.06799. 73

Erick Rodríguez-Hernandez, Juan Irving Vasquez-Gomez, and Juan Carlos Herrera-Lozada. Flying through gates using a behavioral cloning approach. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1353–1358, 2019. doi: 10.1109/ICUAS.2019.8798172. 18

L.B. Rosenberg. Virtual fixtures: Perceptual tools for telerobotic manipulation. In *Proceedings of IEEE Virtual Reality Annual International Symposium*, pages 76–82, 1993. doi: 10.1109/VRAIS.1993.380795. 88

Adam Santoro, Sergey Bartunov, Matthew M. Botvinick, Daan Wierstra, and Timothy P. Lillicrap. One-shot learning with memory-augmented neural networks. *CoRR*, abs/1605.06065, 2016. URL http://arxiv.org/abs/1605.06065. 15

Ravi Kiran Sarvadevabhatla, Jogendra Kundu, and Venkatesh Babu R. Enabling my robot to play pictionary: Recurrent neural networks for sketch recognition. MM '16, page 247–251, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450336031. doi: 10.1145/2964284.2967220. URL https://doi.org/10.1145/2964284.2967220. 64

Fumihiro Sasaki and Ryota Yamashina. Behavioral cloning from noisy demonstrations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=zrT3HcsWSAt. 17

Tomoya Sasaki, MHD Saraiji, Charith Lasantha Fernando, Kouta Minamizawa, and Masahiko Inami. Metalimbs: multiple arms interaction metamorphism. In *ACM SIGGRAPH 2017 Emerging Technologies*, page 16. ACM, 2017. 19, 21, 24

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 54

Siri. https://machinelearning.apple.com/research/siri-voices, 2017. 67

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning. *CoRR*, abs/1703.05175, 2017. URL http://arxiv.org/abs/1703.05175. 15

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15:1929–1958, 2014. 58

Susumu Tachi, Yasuyuki Inoue, and Fumihiro Kato. Telesar vi: Telexistence surrogate anthropomorphic robot vi. *International Journal of Humanoid Robotics*, 17, 08 2020. doi: 10.1142/S021984362050019X. 12

Shintaro Tada, Kohei Sonoda, and Takahiro Wada. Simultaneous achievement of workload reduction and skill enhancement in backward parking by haptic guidance. *IEEE Transactions on Intelligent Vehicles*, 1(4):292–301, 2016. doi: 10.1109/TIV.2017.2686088. 88

A. Takagi, G. Xiong, H. Kambara, and Y. Koike. Endpoint stiffness magnitude increases linearly with a stronger power grasp. *Scientific Reports*, 10(1), January 2020. doi: 10.1038/s41598-019-57267-0. URL https://doi.org/10.1038/s41598-019-57267-0. 38

Atsushi Takagi, Gowrishankar Ganesh, Toshinori Yoshioka, Mitsuo Kawato, and Etienne Burdet. Physically interacting individuals estimate the partner's goal to enhance their movements. *Nature Human Behaviour*, 1(3), 2017. doi: 10.1038/s41562-017-0054. URL https://doi.org/10.1038/s41562-017-0054. 38

Atsushi Takagi, Francesco Usai, Gowrishankar Ganesh, Vittorio Sanguineti, and Etienne Burdet. Haptic communication between humans is tuned by the hard or soft mechanics of interaction. *PLOS Computational Biology*, 14(3):e1005971, March 2018. doi: 10.1371/journal.pcbi.1005971. URL https://doi.org/10.1371/journal.pcbi.1005971. 38

K.P. Tee, J. Li, L.T.P. Chen, and G. Ganesh. Towards emergence of tool use in robots: Automatic tool recognition and use without prior tool learning. *In proceedings, IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2018. 21

Chakkrit Termritthikun, Yeshi Jamtsho, and Paisarn Muneesawang. An improved residual network model for image recognition using a combination of snapshot ensembles and the cutout technique. *Multimedia Tools and Applications*, 79(1-2):1475–1495, November 2019. doi: 10.1007/s11042-019-08332-3. URL https://doi.org/10.1007/s11042-019-08332-3. 59

Alexander Toet, Irene A. Kuling, Bouke N. Krom, and Jan B. F. van Erp. Toward enhanced teleoperation through embodiment. *Frontiers in Robotics and AI*, 7, February 2020a. doi: 10.3389/frobt.2020.00014. URL https://doi.org/10.3389/frobt.2020.00014. 42

Alexander Toet, Irene A. Kuling, Bouke N. Krom, and Jan B. F. van Erp. Toward enhanced teleoperation through embodiment. *Frontiers in Robotics and AI*, 7, February 2020b. doi: 10.3389/frobt.2020.00014. URL https://doi.org/10.3389/frobt.2020.00014. 12, 62

J. Ventre-Dominey, G. Gibert, M. Bosse-Platiere, A. Farnè, P. F. Dominey, and F. Pavani. Embodiment into a robot increases its acceptability. *Scientific Reports*, 9(1), July 2019. doi: 10.1038/s41598-019-46528-7. URL https://doi.org/10.1038/s41598-019-46528-7. 11, 62

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, October 2019. doi: 10.1038/s41586-019-1724-z. URL https://doi.org/10.1038/s41586-019-1724-z. 17

Niklas Wahlström, Thomas B. Schön, and Marc Peter Deisenroth. From pixels to torques: Policy learning with deep dynamical models, 2015. URL https://arxiv.org/abs/1502.02251. 18

Daniel S Walker, Robert P Wilson, and Gunter Niemeyer. User-controlled variable impedance teleoperation. In *2010 IEEE International Conference on Robotics and Automation*. IEEE, May 2010. doi: 10.1109/robot.2010.5509811. URL https://doi.org/10.1109/robot.2010.5509811. 9, 38, 40

Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images, 2015. URL https://arxiv.org/abs/1506.07365. 18

Bernhard Weber, Ribin Balachandran, Cornelia Riecke, Freek Stulp, and Martin Stelzer. Teleoperating robots from the international space station: Microgravity effects on performance with force feedback. 11 2019. doi: 10.1109/IROS40897.2019.8968030. 6

Trent Weiss and Madhur Behl. Deepracing: Parameterized trajectories for autonomous racing. 2020. doi: 10.48550/ARXIV.2005.05178. URL https://arxiv.org/abs/2005.05178. 17

Daniel M. Wolpert, Jörn Diedrichsen, and J. Randall Flanagan. Principles of sensorimotor learning. *Nature Reviews Neuroscience*, 12(12):739–751, October 2011. doi: 10.1038/nrn3112. URL https://doi.org/10.1038/nrn3112. 41, 42

Jiajun Wu, Ilker Yildirim, Joseph J. Lim, Bill Freeman, and Joshua B. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28 (NIPS)*, pages 127–135. Curran Associates, Inc., 2015. URL http://papers.nips.cc/paper/5780-galileo-perceiving-physical-object-properties-by-integrating-a-phy pdf. 18

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. URL http://arxiv.org/abs/1609.08144. 73

Dongbo Xiao and Roger Hubbold. Navigation guided by artificial force fields. In *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '98*. ACM Press, 1998. doi: 10.1145/274644.274671. URL https://doi.org/10.1145/274644.274671. 88

Chenguang Yang, Gowrishankar Ganesh, Sami Haddadin, Sven Parusel, Alin Albu-Schaeffer, and Etienne Burdet. Human-like adaptation of force and impedance in stable and unstable interactions. *IEEE Transactions on Robotics*, 27(5):918–930, October 2011. doi: 10.1109/tro.2011.2158251. URL https://doi.org/10.1109/tro.2011.2158251. 38

Shumei Yu, Jiateng Wang, Jinguo Liu, Rongchuan Sun, Shaolong Kuang, and Lining Sun. Rapid prediction of respiratory motion based on bidirectional gated recurrent unit network. *IEEE Access*, 8:49424–49435, 2020. doi: 10.1109/access.2020.2980002. URL https://doi.org/10.1109/access.2020.2980002. 64

Heiga Zen and Andrew Senior. Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3844–3848, 2014. doi: 10.1109/ICASSP.2014.6854321. 67

Tengteng Zhang and Hongwei Mo. Reinforcement learning for robot research: A comprehensive review and open issues. *Int. J. Adv. Robot. Syst.*, 18(3):172988142110073, 2021. 15

You Zhou, Jianfeng Gao, and Tamim Asfour. Movement primitive learning and generalization: Using mixture density networks. *IEEE Robotics Automation Magazine*, 27 (2):22–32, 2020. doi: 10.1109/MRA.2020.2980591. 68