Robust visual detection and tracking of complex objects : applications to space autonomous rendez-vous and proximity operations

Antoine Petit

▶ **To cite this version:**

Antoine Petit. Robust visual detection and tracking of complex objects : applications to space autonomous rendez-vous and proximity operations. Computer Vision and Pattern Recognition [cs.CV]. Université de Rennes, 2013. English. NNT : 2013REN1S190 . tel-00931604v2

HAL Id: tel-00931604

https://theses.hal.science/tel-00931604v2

Submitted on 21 Aug 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ANNÉE 2013

UNIVERSITÉ DE RENNES 1

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Européenne de Bretagne

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Traitement du Signal et Télécommunications

École doctorale Matisse

présentée par

Antoine Petit

préparée à l'unité de recherche Inria Rennes - Bretagne Atlantique
Institut de Recherche en Informatique et Systèmes Aléatoires

Robust visual detection and tracking of complex objects:

applications to space autonomous rendezvous and proximity operations

Thèse soutenue à Rennes le 19 décembre 2013
devant le jury composé de :

Patrick Bouthemy
Directeur de recherche Inria, Inria Rennes-Bretagne Atlantique/président du jury

Frédéric Jurie
Professeur, Université de Caen/rapporteur

Simon Lacroix
Directeur de recherche CNRS, LAAS/rapporteur

Giorgio Panin
Research Associate, German Aerospace Center (DLR)/examinateur

Keyvan Kanani
Ingénieur, EADS Astrium/examinateur

Éric Marchand
Professeur, Université de Rennes 1/directeur de thèse

Acknowledgments

To my jury

First of all I would like to thank my thesis committee, starting with Patrick Bouthemy, who honored me as the chairman. My thanks also go to my rapporteurs Frédéric Jurie and Simon Lacroix, for having accepted to read and review my works, with their proficiency and expertise, in the computer vision and robotics communities. I thank them for attending the defense, despite their distant home bases, and for their questions, on technical or more general concerns. I also would like to thank Giorgio Panin, whose previous works inspired mines and who traveled from Germany to attend the defense, with a deep and appreciable interest. My warmest thanks also go to Keyvan Kanani, who has been supervising me, from the Astrium side, for almost four years from the initial internship preceding the PhD. His regular follow-up, his patience, his advices and his pragmatism and his willingness to extract technological transfers from my research towards Astrium and the industry, along with his constant kindness, during the numerous teleconferences or the visits at Astrium, have been very beneficial and pleasant and have enabled a rich collaboration with Astrium, with further prospects.

I want to deeply thank Éric Marchand, my PhD director. Through his scientific supervision, his patience, his sincere and reassuring advices and discussions, and through the freedom and autonomy he has given me for my works, the achievement of this PhD has been blossoming. I have also appreciated his taste for the aerospace domain and the discussions that have ensued from that. Finally, his supervision and reviews of this thesis, and his help for the preparation for the defense were essential.

To my family

These three years as a PhD student have not been as calm and charming as the Vilaine and my family has been an essential support, constant although distant, to keep the boat afloat, at each level, in all conditions, despite sporadic communications from Armoric. I want to deeply thank my dear parents, grand parents, godmother and godfather, brother and sister, brother and sister in law, and my future nephew/niece. My thanks also go to my cousin Elizabeth for her very kind support for the defense.

Wink to Lagadic

The professional environment in which I have achieved this PhD has been stimulating, peaceful, and friendly, and in this context the Lagadic team has thus been a real "cocoon". First I would like to thank François Chaumette, for his wise follow-up of my works, and for his fair and human management of the team. My warmest thanks also go to Céline Gharsalli, for her precious assistance despite my sometimes chaotic organization, and for her constant kindness. Kindness would also characterized Fabien Spindler, as well as calm and professionalism. I want to thank him for his advices and technical help and rigor. The enthusiasm, fun and sincerity of Alexandre will always be pleasing. I want to pay tribute to Paolo's laugh and to thank him for all these fierce tennis matches, despite this unfair outcome ;-).

The team regularly changes numerically but its spirit remains and to the bunch of PhD students, postdocs, engineers and interns I have met I want address my acknowledgments:

I thus think of my successive office room mates: Guillaume Caron, cheerful bro as a pro and for the cool, the grand Bertrand, for all his Linux hints that were not for deluxe, for his witty-minded mood in front of my absent-minded mood, for all these interminable sport debates and indispensable top tens. Besides his technical help, the joy in every circumstance of terrific Rafik was a chance, wishing him the best for his defense.

With no troll, I think of Aurélien Yol, jolly fellow who will always undergo nutmegs, boon companion under the sun of Rennes, even if it is not worth the Ardennes. For the defense poster, big up! you deserved the cup!

Of course I think of Laurent "el Papillon" Coutard, for his feel good cocktail: daily good common sense, daily good advices and daily good mood, diurnal or nocturnal. If I missed his company during the last year, as well as his epic taste for the aeronautic thing, as a true friend he kept providing me his support and gentleness.

I want to thank Mani for his infinite kindness, wishing to visit him in India soon. My thanks also go to François Pasteau for his listening and experimented advices.

I think of the former mates: Olivier, who I warmly thank for attending my defense, Céline and Caroline, and their feminine kindness, good fellow Clément, crazy Filip, whom I wish the wildest adventures on the road, François Chapeau, Hideaki and more formerly Romain Talloneau, Nicolas Melchior, and Andrea Cherubini and the surf sessions in Brittany.

I also would like to hail the rising and cosmopolite new generation: Riccardo and Giovanni, from the Italian diaspora, Le Cui and his constant sympathy, Nicolas, Vishnu, Suman, and Aly. I thank them for the help during the preparation of the defense. Future is assured!

To Astrium

This thesis was born from a collaboration between Lagadic and Astrium and the follow-up from the people from Astrium has been very professional and pleasant. Besides Keyvan, I would like to deeply thank Noëlla Despré for fer advices and kindness and Roland Brochard, for Surrender and for his very precious help on some programming aspects.

To my friends

My thanks go to my friends who have had the bravery to support and distract me throughout these years. To the fellows from Rennes, thank you for this rock and roll real camaraderie, which could joyfully refresh myself. I especially think of Pauline and Florence who pleased me by attending my defense. I also think of Ioana, who I wish the strongest courage in completing her PhD, and of Josip and his good plans for concerts.

Finally I want to express my gratitude to the buddies from Paris or elsewhere, whom I have not seen much for the last months/years, whom I have not given much news and who have always shown their support and friendship. I especially want to warmly thank Thomas and Hughes for the huge pleasure they favored me with by attending the defense.

Remerciements

À mon jury

En premier lieu je tiens à remercier mon jury de thèse, à commencer par Patrick Bouthemy, qui m'a fait l'honneur de le présider. Mes remerciements s'adressent à mes rapporteurs, Frédéric Jurie et Simon Lacroix, pour avoir accepté de lire et d'évaluer le manuscrit, avec leur grande expertise dans les communauté vision et robotique. Je les remercie également de s'être déplacé pour ma soutenance et pour leurs nombreuses questions sur des aspect techniques ou plus généraux, avec un intérêt qui m'est très valorisant. Je remercie également Giorgio Panin, dont les travaux à TUM ont été une source d'inspiration et qui s'est déplacé d'Allemagne pour participer à la soutenance, avec un intérêt certain et une précision très appréciables. Je remercie aussi très chaleureusement Keyvan Kanani, qui m'a accompagné, côté Astrium, pendant près de quatre ans depuis le stage précédent la thèse. Son suivi régulier, son écoute, sa patience, ses conseils, son pragmatisme, et son engouement pour faire fructifier les travaux réalisés et pour en tirer des transferts technologiques vers Astrium et l'industrie, ainsi que sa sympathie de tous les instants, que ce soit au cours des nombreuses visio/télé conférences ou de mes séjours visites à Toulouse, m'ont été très bénéfiques et agréables, et ont permis une collaboration riche et porteuse avec Astrium.

Je tiens à vivement remercier Éric, mon directeur de thèse. Par son encadrement scientifique, sa patience, ses conseils et discussions sincères et rassurants, et par la liberté et l'autonomie qu'il a pu me donner dans mon travail, l'accomplissement de cette thèse a été un épanouissement, dans une relation doctorant/encadrant équilibrée. J'ai également aprécié son entrain pour les applications dans le domaine spatial et les discussions qui ont pu en découler. Enfin, sa supervision et ses relecture lors de la rédaction du manuscrit, d'articles, ou lors de la préparation de la soutenance m'ont été indispensables.

À ma famille

Ces trois ans de thèse n'ont pas toujours été tranquilles et beaux comme la Vilaine et ma famille a été le soutien essentiel, constant bien que distant, pour maintenir le rafiot à flots, à tous niveau, en toutes conditions, et en dépit de communications souvent sporadiques depuis l'Armorique. Je remercie ainsi mes très chers parents, grands parents, mon parrain, mon frère et ma soeur, beau-frère et belle-soeur, et mon/ma futur/e neveu/nièce. Un grand merci aussi à ma cousine Elizabeth de m'avoir apporté son très gentil soutien pour la soutenance.

Clin d'oeil à Lagadic

L'environnement professionnel dans lequel j'ai pu évolué pendant cette thèse a été à la fois porteur, paisible, franc et amical, et le "cocon" Lagadic en a été le foyer. Je remercie ainsi François Chaumette, pour son oeil avisé sur mes travaux et pour son encadrement toujours juste et humain de l'équipe. Je tiens à remercier Céline Gharsalli, pour son assistance précieuse face à mon organisation chaotique, et pour sa bienveillance continue. Gentillesse, calme et professionnalisme caractériseraient bien Fabien, que je remercie pour ses conseils et sa rigueur technique. L'enthousiasme et la franchise d'Alexandre feront toujours plaisir. Je salue Paolo, pour son rire et ces parties de tennis acharnées, en dépit de cette issue injuste ;-)

L'effectif change régulièrement, mais l'esprit lui perdure, et à la palanquée de camarades doctorants, postdocs, ingénieurs, stagiaires que j'ai pu cotoyer, croiser à Lagadic, j'adresse mes hommages, parce que sous les claviers, la plage:

Je pense ainsi à mes "co-bureaux" successifs, Guillaume Caron, joyeux larron de bureau et de sortie, au grand Bertrand, pour ses coups de mains Linux qui n'étaient pas du luxe, pour son humour face à mes humeurs, entre débats sportifs interminables, et top ten matinaux indispensables. En plus de son aide technique, la gaieté en toute circonstance de Rafik le magnifique a été une chance, en lui souhaitant par ailleurs le meilleur pour sa soutenance. Sans faire le mariolle, je pense à Aurélien Yol, gai luron qui se prendra toujours des petits ponts, fidèle compère sous le soleil de Rennes, même si ça ne vaut pas les Ardennes. Pour l'affiche gros big up, tu méritais la cup! Je pense bien-sur à Laurent "el Papillon" Coutard et à son cocktail bonheur: bon sens, bons conseils, et bonne humeur au quotidien, diurne comme nocturne. Si sa compagnie m'a manqué une année, comme son gout épique pour la chose aéronautique, en bon ami il m'a maintenu son précieux soutien et sa bonhomie. Je songe aussi à Mani pour sa gentillesse infinie, et à qui j'espère rendre visite en Inde un de ces jours. Merci à François Pasteau pour son écoute et ses bons conseils .

Je songe aux anciens: à Olivier, qui m'a fait le plaisir d'assister à ma soutenance, à Céline et Caroline et à leur bienveillance féminine, au bon compère Clément, à crazy Filip, à qui je souhaite les plus belles aventures sur la route, à François Chapeau, à Hideaki et plus antérieurement à Romain Talloneau, Nicolas Melchior, et Andrea Cherubini, pour les sessions de surf bretonnes. Je tiens aussi à saluer la génération montante et cosmopolite: Riccardo et Giovanni de la diaspora italienne, Le Cui et sa constante sympathie, Nicolas le bon poulain, Vishnu, Suman, Aly. Je les remercie pour leur aide pendant la préparation de la soutenance. La relève est assurée!

À Astrium

Cette thèse est née d'une collaboration entre Lagadic et Astrium et le suivi des équipes d'Astrium a été très professionnel et sympathique. Outre Keyvan Kanani, je tiens à remercier Noëlla Despré pour ses conseils et sa gentillesse, ainsi que Rolland Brochard, pour Surrender et pour son aide très précieuse sur mon code.

À mes amis

Je vais remercier mes amis qui on eu le courage de me suivre de près ou de loin. Aux rennais, merci pour cette camaraderie rock'n'roll qui m'a bien changé les idées. Je pense notamment à Pauline et à Florence pour être venu me soutenir le 19 décembre. Je pense aussi à Ioana, à qui je souhaite plein de courage dans la dernière ligne droite de sa thèse, et à Josip, notamment pour les bons plans concerts. Je pense enfin aux compères de Paris ou d'ailleurs, que je n'ai pas beaucoup vu ces derniers mois/années, à qui je n'ai pas donné beacoup de nouvelles mais qui m'ont toujours manifesté leur soutien et leur amitié. Je remercie spécialement Thomas et Hugues, qui m'ont fait l'immense plaisir de se déplacer depuis Paris pour la soutenance.

Contents

Notations

General mathematics

a	:	scalar
$\mathbf{a}$	:	vector
$\mathbf{A}$	:	matrix

Algebra

$\mathbb{R}^n$	:	real space with n dimensions
$\mathbb{E}^n$	:	Euclidean space with n dimensions
$\mathbb{P}^n$	:	projective space with n dimensions
$SO(3)$	:	Special Orthogonal group
$SE(3)$	:	Special Euclidean group
$\mathbf{A}^\top$	:	transpose of matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	:	inverse of matrix $\mathbf{A}$
$\mathbf{A}^+$	:	pseudo-inverse of matrix $\mathbf{A}$
$tr(\mathbf{A})$	:	trace of matrix $\mathbf{A}$
$diag(\mathbf{a})$	:	diagonal matrix with coefficient $\mathbf{a}$
$[\mathbf{v}]_\times$	:	skew symmetric matrix related to vector $\mathbf{v}$
$\mathbf{I}_n$	:	identity matrix with $n \times n$ dimensions

Geometry

$\mathcal{X}$	:	point in $\mathbb{E}^3$
$\mathcal{F}_a$	:	Cartesian coordinate frame a
${}^a\mathbf{X} = \begin{bmatrix} {}^aX & {}^aY & {}^aZ \end{bmatrix}^T$	:	vector representing the coordinates of point $\mathcal{X}$ in $\mathcal{F}_a$
${}^a\overline{\mathbf{X}} = \begin{bmatrix} \lambda {}^a\mathbf{X}^T & \lambda \end{bmatrix}^T = \begin{bmatrix} {}^a\mathbf{X}^T & 1 \end{bmatrix}^T$	:	vector representing the homogeneous coordinates of point $\mathcal{X}$ in $\mathcal{F}_a$
${}^b\mathbf{M}_a = \begin{bmatrix} {}^b\mathbf{R}_a & {}^b\mathbf{t}_a \\ \mathbf{0} & 1 \end{bmatrix}$	:	homogeneous matrix describing the 3D rigid transformation in $\mathbb{P}^3$ from $\mathcal{F}_a$ to $\mathcal{F}_b$
${}^b\mathbf{R}_a = exp([\theta\mathbf{u}]_\times)$	:	rotation matrix
${}^b\mathbf{t}_a$	:	translation vector
$\theta\mathbf{u}$	:	rotation vector
$\mathbf{x}$	:	metric coordinates of a 2D point in the image
$\overline{\mathbf{x}} = \begin{bmatrix} \lambda\mathbf{x}^T & \lambda \end{bmatrix}^T$	:	metric homogeneous coordinates of a 2D point in the image
$\mathbf{p}$	:	pixel coordinates of a 2D point in the image
$\overline{\mathbf{p}} = \begin{bmatrix} \lambda\mathbf{p}^T & \lambda \end{bmatrix}^T$	:	pixel homogeneous coordinates of a 2D point in the image

Introduction

The expansion of autonomous systems in technology, industry and in our daily life, reflects our propensity to face new challenging issues. These issues can refer to complex or dangerous tasks humans cannot do but also to simple or labored tasks humans are reluctant to do.

Autonomy can be defined by the ability of a system to act on its own in its environment and to interact with this environment. It is a central concept in the fields of artificial intelligence or robotics. Fostered or imagined by visionaries such as Alan Turing or Isaac Asimov, these branches have constantly improved from the second half of the 20^{th} century. In popular culture, the chess-playing computer *Deep blue* or the *Honda P-series* humanoid robots, have been famous and pioneering examples of such intelligent and autonomous systems.

In robotics, these systems, having gained sufficient maturation, are currently flourishing in a wide range of applications, for scientific, exploration, industrial or domestic concerns. In the automotive industry, the *Google* driverless system is a famous example. For medical applications, the boom of robotics has been demonstrated by the *Intuitive Surgical*'s *da Vinci* semi-autonomous surgical system. In the military industry, autonomous Unmanned Aircraft Vehicles (UAV), such as the *Northrop Grumman*'s *X-47B*, are actively developed and tested. The successes of the Mars exploration rovers *Spirit*, *Opportunity* and more recently *Curiosity*, operated by the NASA's *Jet Propulsion Laboratory*, have been milestones for autonomous robotic systems in the aerospace field. Finally, the domestic area is now being flood by robots, like robotic vacuum cleaners or grass mowers.

Sensing the world: the key to autonomy

An autonomous robotic system can be generally modeled as a set of sensors to perceive the environment, and a control system to drive a set of actuators and end-effectors, to accomplish tasks such as grasping objects, walking, rolling or flying, the whole working in closed loop. In this way, such a system differentiate itself from tele-operated systems. In order, for a robotic system, to autonomously and properly achieve its goal and to adapt to the environment, sensing and providing reliable information to the control system is a crucial issue, especially in changing environments. Numerous sensors are available, working globally, like Global Positioning System (GPS), or relatively to the local environment such as camera or range sensors, or relatively to the system itself like Inertial Measurements Units (IMU, gyroscopes and accelerometers). They can work actively or passively, whether the sensor interacts with the environment (force or touch sensors, range sensors,

for instance LIDAR, RADAR or SONAR) or not (visible or thermal cameras). Among them, vision sensors and monocular cameras are particularly popular and widespread.

Monocular cameras have indeed several advantages that make them applicable for many robotic applications. By capturing images though the detection of electromagnetic radiations, cameras provide a reliable and stable 2D visual information of the environment, wether it is indoor, outdoor, underwater, aerial, space, etc. The Field of View (FoV) can be tuned and potentially be omnidirectional. They are relatively cheap facilities, with respect to Inertial Measurement Units (IMU), GPS or range sensors for instance. They can have a small form factor, and can thus be conveniently mounted on many kinds of systems. Besides, they require few power consumption and their calibration process (for conventional cameras) is fast and easy. For these reasons, which regard hardware aspects, cameras can be found on mobile robots, UAVs, underwater robots, industrial robot arms, surgical robots, space robots, etc.

The images captured by the camera need to be understood to provide an exploitable information for the control system, through localization in the environment for instance. These concerns, related to software aspects, refer to the field of computer vision.

"Vision is the art of seeing what is invisible to others" - Jonathan Swift

Computer vision aims at processing, analyzing and understanding the content of images.The development of cameras, webcams, and the growing production of image data and videos, have considerably increased the need for softwares and algorithms to process these images, making computer vision a particularly active research field.

Generally speaking, with computer vision, different issues can be targeted. Based on an image or on a set of images, resulting from video sequences, acquired by one or multiple cameras, a computer vision system has to deal with detecting, recognizing, tracking or reconstructing patterns, scenes, objects, people from images. The overall goal of these tasks is to understand the scene or localize it, with potentially complex environments or imaging conditions. It relies on machine learning, pattern recognition, motion analysis techniques and on the basic idea of fitting the image content with some predefined or learned model. In addition to robotics, other applications, such as classification and indexation of images and augmented reality, are related to computer vision.

Objective of this thesis

This thesis has come within this scope of computer vision for robotics. More specifically, the main objective of this work has been to design computer vision solutions able to localize a camera, and the potentially underlying robotic system, with respect to a known object, and with a particular focus on space robotics and space rendezvous applications. In this case, the camera would be mounted on a space robotic vehicle, aiming at approaching a target space object or a spacecraft.

Providing a high-level of autonomy to robotic systems, and relying on computer vision technologies for this purpose, is particularly suitable for space robotics applications, especially to the case of space autonomous rendezvous or proximity operations.

Computer vision solutions have already been successfully implemented on some space robotic systems, as shown by the Mars rovers Spirit or *Curiosity*. In the context of space rendezvous, the idea of conceiving fully autonomous Guidance Navigation and Control

(GNC) systems for a space vehicle with respect to its target, is preferable, but also very challenging. Few operational or experimental systems have indeed been designed and tested in this sense. For servicing goals, we can mention the *Progress* spacecraft or the Automated Transfer Vehicle (ATV), for servicing purposes with the International Space Station (ISS), as some of the rare examples. In the case of space debris removal applications, the problem is novel. Besides, such rendezvous maneuvers have very strict requirements in terms of navigation measurements, and imaging conditions are specific and can be difficult, with potential important light/dark or specular effects, noise...

With the support of Fondation EADS, this thesis was born out of a collaboration between Astrium Toulouse, an aerospace subsidiary of the European Aeronautic Defence and Space Company (EADS) which provides space systems such as space launchers and satellites, and the Lagadic research team at Inria Rennes, whose research topics involve computer vision, visual servoing and robotics.

Contributions of this thesis

The aim of this work has been to provide a unified computer vision-based localization system, particularly in the context of a space autonomous rendezvous. This system should be able to estimate the full 6 Degrees of Freedom (DoF) localization parameters (position and orientation), which refer to the so-called full pose, between the camera and the considered target. Let us remind that the target is assumed to be known. More precisely, our solution is based on the prior knowledge of a fixed 3D CAD model of the target. We have paid attention to three major aspects of the problem:

- **Visual detection and initial localization of the target**

  A first contribution has been to work out a novel solution to initially determine the complete pose between the camera and the target object. As previously mentioned, our method relies on the prior knowledge of the 3D CAD model of the target. This solution is based on segmenting the target from its background on a set of initial input images, through the development of a **foreground/background segmentation** technique particularly suited for our context. The silhouettes of the segmented images are then used for matching a set of pre-generated synthetic views of the target (using the 3D model) with the initial input images. In order to accomplish this task, a **probabilistic matching and alignment framework** between the views and the image has been designed to retrieve the full 6 DoF pose. For a better computational efficiency, we have developed a solution to classify the synthetic views into a hierarchical view graph.

- **Visual tracking of the target**

  The second issue focuses on estimating the complete pose of the camera with respect to the target through frame-by-frame model-based tracking, initialized by the detection technique. We first propose a method able to deal with **complex 3D models**, by taking advantage of rendering capacity and hardware acceleration. Another

contribution has then been to robustly **combine complementary sorts of visual information**. We have chosen visual features to represent the target object through its edges, its shape and its texture, using classical edge-based, color-based and interest point based features. The pose estimation process is then based on the minimization of an error function with respect to the pose. We show the efficiency and robustness of the implemented solution, quantitatively on synthetic data, and qualitatively on real image sequences.

- **Measuring the uncertainty of the localization process**

  We suggest to evaluate the reliability of the tracking process by propagating uncertainty from the visual features to the camera displacement. Based on this uncertainty, a **Kalman filtering** process and **pose prediction** scheme has been designed for the tracking method, to smooth pose estimates, to provide to handle potential large inter-frame motions.

Outline of the thesis

This thesis is organized as follows. Chapter 1 presents the application field of this thesis, which is space autonomous rendezvous and proximity operations, for on-orbit servicing and debris removal goals. The issues regarding navigation (or localization) are particularly stressed out. The reasons for relying on computer vision in this context are explained and the related works are described. The experimental procedures, inherent to this application, and which aims at validating the implemented solutions, are also presented.

Chapter 2 introduces some theoretical background on computer vision, focusing on two issues related to our problem. The first one concerns visual initial localization of the target in the image, through pose estimation by detection. The second one tackles the issue of visual localization of the target, based on pose estimation by frame-by-frame tracking.

Chapter 3 deals with the solution we propose to handle the issue of detection and initial pose estimation, providing the technical aspects as well as experimental results on various data.

In chapter 4, we present our method to address the problem of pose estimation by frame-by-frame tracking, of determining the reliability of the tracking process, with various experimental validations.

Finally, a conclusion recaps the proposed approaches and the obtained results, and some future perspective are suggested.

Publications

International Conferences

[C1] A. Petit, E. Marchand, K. Kanani. Combining complementary edge, point and color cues in model-based tracking for highly dynamic scenes. In *IEEE Int. Conf. on Robotics and Automation, ICRA 2014*, Honk Kong, China, June 2014.

[C2] A. Petit, E. Marchand, K. Kanani. A robust model-based tracker for space applications: combining edge and color information. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'2013*, Tokyo, Japan, November 2013.

[C3] A. Petit, E. Marchand, K. Kanani. Augmenting Markerless Complex 3D Objects By Combining Geometrical and Color Edge Information. In *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR 2013*, page. 287-288, Adelaide, Australia, October 2013.

[C4] A. Petit, E. Marchand, K. Kanani. Tracking complex targets for space rendezvous and debris removal applications. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'12*, page. 4483-4488, Vilamoura, Portugal, October 2012.

[C5] A. Petit, E. Marchand, K. Kanani. Vision-based Detection and Tracking for Space Navigation in a Rendezvous Context. In *Int. Symp. on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS*, Torino, Italia, September 2012.

[C6] K. Kanani, A. Petit, E. Marchand, T. Chabot, B. Gerber. Vision-based navigation for debris removal missions. In *63rd International Astronautical Congress*, Naples, Italia, September 2012.

[C7] A. Petit, G. Caron, H. Uchiyama, E. Marchand. Evaluation of Model based Tracking with TrakMark Dataset. In *2nd Int. Workshop on AR/MR Registration, Tracking and Benchmarking*, Basel, Switzerland, October 2011.

[C8] A. Petit, E. Marchand, K. Kanani. Vision-based Space Autonomous Rendezvous : A Case Study. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'11*, page. 619-624, San Francisco, USA, September 2011.

[C9] A. Petit, N. Despré, E. Marchand, K. Kanani, F. Chaumette, S. Provost, G. Flandin. 3D Model-based Visual tracking for Space Autonomous Rendezvous. In *8th Int. ESA Conf. on Guidance and Navigation Control Systems, GNC'11*, Carlsbad, Czech Republic, June 2011.

Misc

[M1] A. Petit, E. Marchand, K. Kanani. Détection et suivi basé modèle pour des applications spatiales, Congrès francophone des jeunes chercheurs en vision par ordinateur, ORASIS'13, Cluny, France, June 2013.

[M2] K. Kanani, I. Ahrns, A. Petit, T. Chabot, A. Pisseloup, E. Marchand. Vision-based navigation and debris characterization, 2nd European Workshop on Active Debris Removal, CNES, Paris, June 2012.

Chapter 1

Space autonomous rendezvous and proximity operations

For decades, the advances in space exploration and the development of space commercial operations have enhanced the issue of *on-orbit servicing*. Providing maintenance, supplying, refueling, repairing of space facilities and removing space debris have become key requirements to efficiently and reliably pursue space exploration and to maintain a sustainable space environment. The famous successes of the repair missions of the Space telescope Hubble and the assembly and re-supply of the International Space Station, as well as numerous other projects and demonstrators, have been promising examples for facing some of these crucial challenges. In terms of technology, these critical operations require reliable performances of rendezvous and proximity operations between the servicing vehicle (the chaser), and the serviced one or the debris, referred as the target. Particularly, the incorporation of autonomy regarding guidance, navigation and control (GNC) of the chaser with respect to the target during rendezvous, berthing, docking maneuvers, has increasingly crystallized efforts, in order to cope with impossible or limited ground control, due to large communication delays and for operational efficiency reasons. With this growing and key need for autonomous on-orbit operations, works and researches have focused on designing navigation solutions consisting in estimating the relative state between the chaser and the target or debris. For these purposes, the maturation of computer vision technologies for robotics has made them studied and experimented solutions.

The objective of this chapter is to set the scope and the application field of this thesis. An exhaustive review of the issues and concepts related to on-orbit servicing and space autonomous rendezvous and proximity operations can be found in [Fehse 08] and [NASA 10]. Some basic concepts regarding the whole process of a space rendezvous mission will be presented in section 1.1. Some particular attention will be paid on proximity operations and on the final phase of a rendezvous (section 1.2), with a focus on the possible navigation solutions. Finally, section 1.3.4 discusses how computer vision and robotic techniques can be considered and specifies the consequent requirements, especially with regards to the solution proposed in this thesis.

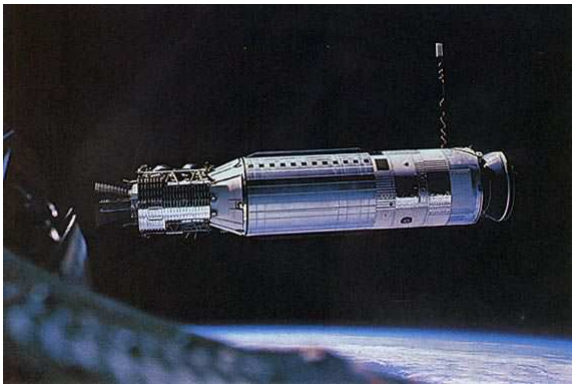
1.1 Space rendezvous

1.1.1 Some principles, a brief history and current stakes

A space rendezvous consists in a series of successive space orbital maneuvers which aim at bringing an active spacecraft (the chaser) in the vicinity or in contact with another spacecraft or object (the target). This process is currently frequently used for different kinds of space missions: re-supply of orbital stations, exchange of crew in orbital stations, inspection and repair of spacecrafts. Along history and since March 16^{th} 1966 and the rendezvous and docking between Gemini and an Agena target vehicle (Figure 1.1(a)), the world has witnessed many famous successful examples of such maneuvers and missions, carried out for experimental or operational purposes, essentially by Russian, US and more recently European, Japanese and Chinese space programs:

- In 1967 experimental Soviet vehicles Cosmos 186 and 188 docked, for what is known as the first autonomous rendezvous.
- Soyuz 4 and Soyuz 5 performed the first rendezvous and docking with an exchange of crew, in 1969.
- With Salyut and Mir Space Station Programs (1971-1999), Russian operational manned Soyuz spacecraft, for crew exchange, and unmanned Progress spacecraft, for re-supply, regularly performed autonomous docking on both stations.
- In 1975, Americans and Russian also experienced the first international docking mission, between an Apollo capsule and a Soyuz spacecraft.
- The different US Space Shuttles, designed for Space Transportation System (STS) program conducted by the NASA, started their servicing missions in 1984 with the repair of the Solar Maximum satellite and provided re-suppply and crew exchange missions with the Mir space stations in the 90s, until 1999. Space shuttles continued their missions later on with the International Space Station (ISS) from 1998 until 2011 with the STS-135 mission of the Atlantis Space Shuttle.
- Famous examples of satellite servicing are the successive repairs of Hubble Space Telescope between 1993 and 2009 with four manned missions involving space shuttles initiated by the NASA to repair or replace optical instruments or different deficient sensors or actuators embedded on the telescope.
- Since its assembly, the ISS has been serviced by both Progress spacecrafts for re-supply and Soyuz TMA spacecrafts (Figure 1.1(b)) for crew exchange. More recently, the Automated Transfer Vehicle (ATV, Figure 1.1(c)), operated by the European Space Agency (ESA), and the Japanese H-II Transfer Vehicle (HTV) have successfully delivered supplies to the ISS, with fully (ATV) or partially (HTV) autonomous docking capabilities, respectively since 2008 and 2009.
- Since 2012 cargo supply to the ISS has been also flown out by a privately owned commercial craft, with SpaceX's partially reusable and fully autonomous Dragon spacecraft (Figure 1.1(d)), through berthing and docking maneuvers. This trend

originated by the NASA to rely on private commercial industries has been emphasized by the launch of Orbital Sciences'Cygnus spacecraft, which has joined the fleet of autonomous re-supply vehicles since September 2013.



(a) View of Agena target vehicle from Gemini 8 spacecraft.

(b) Soyuz TMA spacecraft right docking with the ISS.

(c) ESA ATV vehicle.

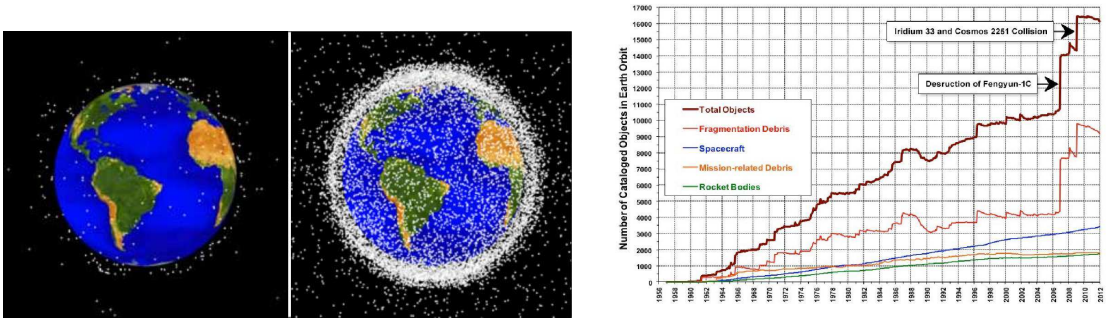
(d) Space X Dragon spacecraft docked with the ISS.

Figure 1.1 – Examples of rendezvous and docking missions.

Several challenges and stakes justify the need for on-orbit servicing and rendezvous capabilities and especially the need for autonomous systems. With the expansion of space activities, for economic and earth observations concerns, with the constant growth of telecommunication and observation satellites, as well as for exploration concerns, as demonstrated by the recent success of the Mars Science Laboratory mission and the future launch of Gaia and James Webb Telescopes, servicing these assets to preserve their functionality and extend their life, through repairing or refueling, would both provide economic and scientific benefits.

Besides, as evoked by [Liou 10, NASA 12, Bonnal 13b], the growth of the population of space objects, in recent years, especially on the Low Earth Orbits (LEO)(between 700km and 1000km), faces saturation (see Figure 1.2). According to these studies, a tipping point, the so-called *Kessler syndrome*, is being reached. The Kessler syndrome, early identified in the 70s by Donald J. Kessler and Burt Cour-Palais [Kessler 78], states that debris resulting from collisions between objects already on-orbit would cascade. As recent

examples which accelerated this phenomenon, we could mention the collision in 2009 between an American telecommunication Iridium satellite with a defunct Russian military Cosmos satellite, and the intentional destruction, with a anti-satellite device, of the Chinese observation satellite Fengyun-1C, both events engendering thousands of debris (see Figure 1.2(b)). The current situation implies important risks for active spacecrafts or platforms, as proven by the recent debris avoidance maneuvers performed by the ISS, and as shown in [Brudieu 12] for the case of the Spot satellite. As a consequence, a need to stabilize the environment, by removing some debris, such as defunct satellites or rocket bodies, becomes crucial. For instance, the NASA Orbital Debris Office [Liou 10] pointed out that at least 5 large debris should be removed per year from now on. As recently discussed at the last ESA Conference on Space Debris in Darmstadt [1], many on-going studies have addressed this problem, which is often referred as Active Derbis Removal (ADR) and which faces technological, but also economical, legal and political issues.



(a) (b)

Figure 1.2 – Artistic view of space objects (a), in 1963 (left) and in 2013 (right), and number of space objects (b) in Low Earth Orbit [NASA 12].

The general scenario [Bonnal 13b] for an ADR would be to perform rendezvous with the debris, to dock on it or capture it, to control it and finally to de-orbit it. De-orbitation would mean, for debris in LEO, to bring them towards random reentry into the Earth's atmosphere for self-destruction and for debris in the Geostationary Orbit (GEO), which also faces similar problems, a solution would be to move the debris to disposal, safe orbits.

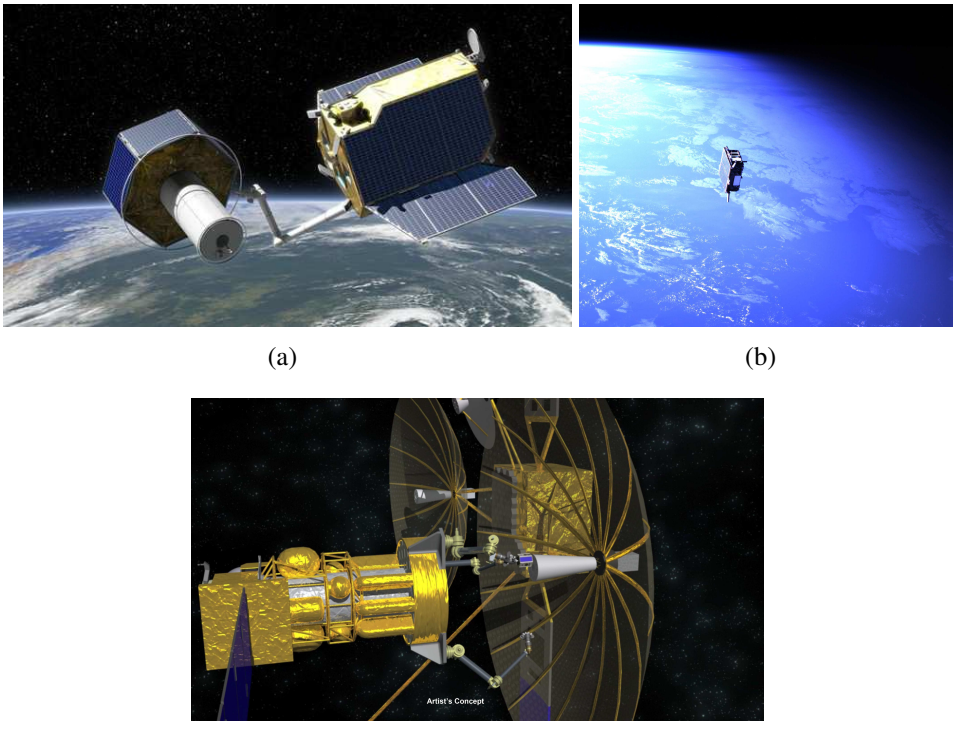
Technically, given the type and location of the considered debris, different strategies and technologies have been stressed out to handle ADR [Bonnal 13b, NASA 12]. The ADR operation could indeed involve a single chaser for a single debris or for several debris, or a swarm of multiple chasers (or kits) for multiple debris. This last idea has been for instance suggested for the Orbit Transfer Vehicle (OTV) program [Martin 13], led by the French Centre National d'Etudes Spatiales (CNES), with Astrium and Thales Alenia Space as contractors.

Among other projects for ADR missions, we can mention the Clean Space Program initiated by ESA[2], with the aim of launching, by 2021, a debris removal vehicle, by first demonstrating its de-orbiting capabilities on a large debris satellite such as Envisat. This program is likely to rely on technologies and platforms studied by the German space

[1] http://congrexprojects.com/2013-events/13a09/introduction
[2] http://www.esa.int/Our_Activities/Space_Engineering/Clean_Space

agency DLR for the DEOS program[3] [Sellmaier 10, Rank 11, Mühlbauer 12]. With Astrium as a contractor, DEOS plans to launch, from 2018, two demonstration satellites, a chaser and a target (see Figure 1.3(a)), to experiment and validate rendezvous, capture and control operations (see also sections 1.1.2.5 and 1.3.4). Such demonstration formation flights are also investigated by the Swedish Space Corporation (SSC) with the PRISMA mission [Persson 06] (see Figure 1.3(b)), co-developed with CNES, DLR and the Danish University of Technology (DTU), for which two satellites were launched in 2010 to test advanced closed-loop formation flying and rendezvous. The Swiss Federal Institute of Technology in Lausanne (EPFL) is developing Clean Space One, with the idea of launching, by 2016, a nanosatellite intended to capture already on-orbit Swiss cube-satellites and to finally ensure reentry into the atmosphere.

Instead of de-orbiting dead satellites or debris, the American Defense Advanced Research Projects Agency (DARPA) has suggested, by initiating the Phoenix program[4] this year, to harvest, reuse or recycle parts of dead satellites into valuable facilities for other missions (Figure 1.3(c)).



(a) (b)

(c)

Figure 1.3 – Artictic view of the DEOS mission (a). Image of the Tango satellite seen from the Mango satellite, for the PRISMA mission (b). Artistic view of the Phoenix program (c).

In terms of technology, both servicing and ADR missions require the achievement of orbital maneuvering, rendezvous and docking, and robotic manipulation. A high level of autonomy would be preferred for these operations. Indeed, due to some telecommu-

[3] http://www.dlr.de/rd/desktopdefault.aspx/tabid-2266/3398_read-36724/
[4] http://www.darpa.mil/Our_Work/TTO/Programs/Phoenix.aspx

nication link constraints, such as limited data rate and delays, ground control and teleoperation are simply impossible in some cases or limited to very simple tasks. Nevertheless, these complex missions, possibly involving uncooperative and tumbling targets, could not handle such constraints, especially on high orbits. Besides, for repairing missions which classically require human in-orbit intervention, as it was the case for Hubble Telescope, Skylab or Solar Maximum missions, autonomy would free human astronauts to perform this risky job.

But servicing spacecrafts and removing debris are not the only fields of applications that the technologies developed for autonomous rendezvous and docking/berthing system would aim at. These technologies are also explored and carried out for autonomous landing of probes on planets or small solar systems bodies, like comets or asteroids, as well as the capture of asteroids or the capture and return of planet samples. The recent successful landing of NASA Mars Science Laboratory on Mars in order to deploy the rover Curiosity in 2012, or the rendezvous and landing of Japanese probe Hayabusa with asteroid Itokawa in 2005 have illustrated the advances of autonomous navigation, control and guidance technologies for such missions.

One particular challenge for these systems especially lies in the nature of the target. For rendezvous with debris or with asteroids, the target is uncooperative and tumbling, with an initially unknown kinematics. This specificity considerably complicates the problem as compared to cooperative rendezvous process, as involved in ISS resupply missions, particularly regarding navigation issues. As it will be reviewed in section 1.3.4, computer vision solutions have been designed and experimented to handle with such complex cases. This is also the objective of the works presented in this thesis.

1.1.2 Phases of a rendezvous mission

Let us first briefly overview the different phases which make of a rendezvous mission and the different related issues, which are more exhaustively reported in [Fehse 08, Woffinden 07, Woffinden 08, Astrium 10, Tingdahl 10]. A rendezvous mission can indeed be divided into five major phases: launch of the spacecraft, phasing, far range rendezvous, close range rendezvous and mating.

1.1.2.1 Launch of the chaser

The launch of the chaser is the operation which brings the chaser from ground until its orbit injection, using a launch system such as rockets. As for the launch of any other space payload, this phase must respect some conditions to be properly performed. First, the location, direction and time of the launch should be finely determined. The launch should indeed occur when the launch site passes through the intended injection orbital plane (see Figure 1.4(a)), which occurs twice a day. But since a launch directed easterly produces gains in terms of velocity, there is only one opportunity per day for a launch, and the tolerated error margin around this time, which implies costly trajectory correction is specified by the launcher capabilities. Once correctly launched, the chaser vehicle is brought into a lower orbit than the target, on the target orbital plane (Figure 1.4(b)), at an arbitrary phase behind the target.

1.1.2.2 Phasing and transfer of the chaser near the target orbit

This step is intended to carefully reduce the phase angle between the chaser and the target and to bring the chaser to a so-called *initial aim point* or *entry gate* on the target orbit or close to it. This step is usually controlled from ground in open loop, with navigation based on absolute measurement (GPS) and consists in some orbital maneuvers following different possible strategies.

In order to present them, let us first define the target Local Orbital Frame (LOF) with axis +V-bar in the direction of the target velocity and -V-bar pointing the ground (Figure 1.4(b)). Most of phasing techniques are based on forward phasing, for which orbital transfer maneuvers are used to bring the chaser from a lower orbit than the target up to the target orbit, in the direction + V-bar (Figure 1.4(b), phase B-C and Figure 1.4(c)), since a lower orbit implies a higher velocity, thus decreasing the phase angle. Backward phasing can also be considered, through drifting from a higher orbit, in the - V-bar direction. The orbital maneuvers generally consist in several perigee and apogee raise maneuvers (Figure 1.4(c)), through tangential impulsive changes of velocity at the apogee or perigee points of the chaser orbit, increasing (apogee) or decreasing (perigee) the eccentricity of the orbit. A famous orbital maneuver is the *Hohmann* transfer, for which apogee and perigee maneuvers are combined to bring the chaser from a circular orbit to another circular orbit closer to the target one. A typical phasing strategy can be seen on Figure 1.4(c), for which the chaser orbit is progressively raised and the phasing rate decreased, until the phase is sufficiently low. A final *Hohmann* transfer is then performed to reach the *initial aim point*, usually at around 30 and 50 km from the target. Another strategy, would be to perform as soon as possible a *Hohmann* transfer to reach a circular orbit very close to the target orbit, and then to achieve successive perigee raise maneuvers to reduce the phasing rate, until some position and velocity conditions are fulfilled, at the so-called *entry gate*. This is safer and desirable when a continuous approach is planned, but requires higher trust capacities to perform this larger *Hohmann* transfer.

1.1.2.3 Far range rendezvous

Phasing maneuvers are operated in open loop and absolute measurements, and result in some uncertainties on the position of the chaser with respect to the target, typically in the order of hundreds of meters in height and a few kilometers in orbital direction. The goal of the far range rendezvous phase is first to acquire the target orbit and position and second to approach the target while reducing relative trajectory uncertainties to respect fine position and velocity conditions to properly start the final close range fully autonomous approach. This objective is achieved with relative measurements, using far range navigation sensors, see section 1.3. Wether both chaser are placed on circular or elliptic orbits, free drift trajectories and tangential and radial orbital maneuvers are used to accomplish the task (Figure 1.4(b), phases D-E and Figure 1.4(c)). With these maneuvers, the phasing rate can be flexibly tuned to ensure synchronizing with a fixed timeline or respect other particular conditions. At the end of this phase, the chaser should usually lie on a hold point on V-bar on the target orbit, at, for instance in the case of the ATV approach, around 2000-1000m from the target, from where the final close range approach shall start.

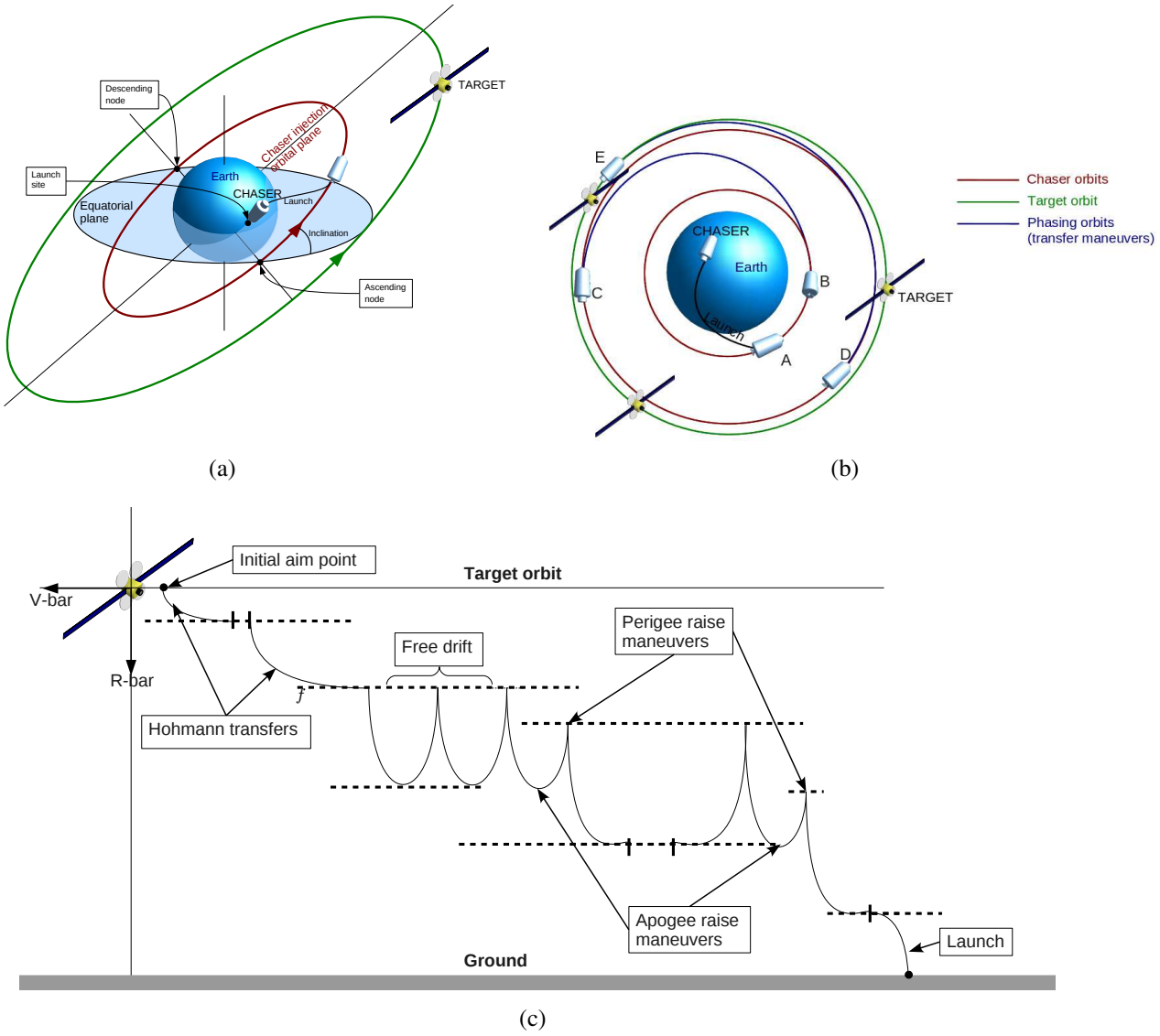


Figure 1.4 – Orbital parameters and phases of a rendezvous mission, in an Earth reference frame (a,b), and in the target local orbital frame (c).

1.1.2.4 Close range rendezvous

Different trajectories and strategies can be considered to perform this approach, which should bring in closed loop the chaser until mating conditions, in terms of relative position, velocities, attitude and angular rates, so that docking or capture tools of the chaser can be safely engaged. For docking, a constant axial velocity should be maintained and for berthing, the chaser should remain in a particular volume so that its manipulator (see section 1.1.2.5) can properly reach the target. Thus, these two rendezvous modalities result in different ending conditions for the close range rendezvous phase. The typical closed loop trajectory for a fully autonomous on-board navigation and control system is a straight line, inside an approach corridor that would ensure safety conditions regarding collision avoidance. A possible fly-around phase within this trajectory can be added to realign the chaser and target docking interfaces (see Figure 1.5).

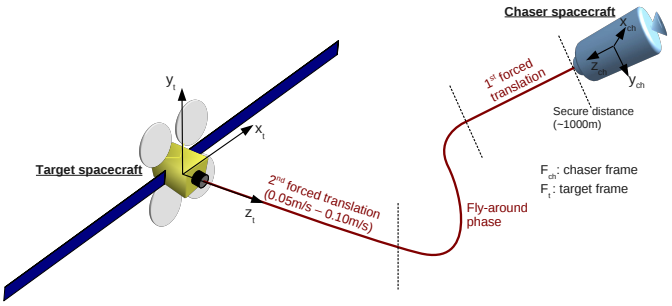


Figure 1.5 – Close range rendezvous - final approach

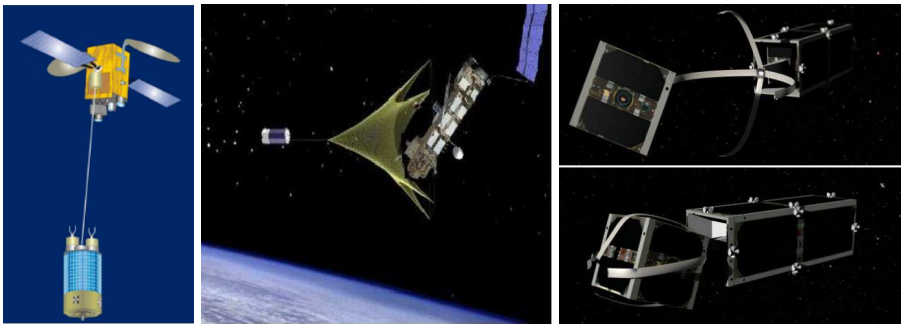
1.1.2.5 Mating: docking or berthing

As introduced before, two different capture mechanisms are usually considered once the chaser is correctly brought in the vicinity of the target. Docking is used so that the chaser actively maneuvers under its own propulsion to connect to the target. Berthing is carried out in order to attach the chaser and the target using a robotic arm, for the final few meters of the rendezvous process. Docking and berthing mechanisms greatly vary given the type of target considered: passive or active, cooperative or uncooperative.

For instance, the cooperative docking systems on the ISS used for the ATV [Pinard 07], the Soyuz and Progress rendezvous missions, consist in the insertion of a probe mounted on the chaser into a passive drogue specially installed on the ISS docking module (see Figure 1.1(b) for the Soyuz spacecraft). This *drogue and probe* mechanism, which was first used for Apollo docking missions to the Apollo lunar module or to Skylab, is also studied for uncooperative and thus passive targets, using for instance the apogee motor nozzle of the target as the drogue in the case of satellite on-orbit servicing, as for the SMART-OLEV project [Kaiser 08], enabling to dock onto many kinds of satellites.

For berthing, the servicing missions of the Space Shuttles used robotic arm Canadarm1 to capture the Hubble Telescope and berth with it for its repairing missions. Since the installation of robotic arm Canadarm2 on the ISS in 2001, the Dragon (Figure 1.1(d)) and HTV supplying vehicles have used it to be grappled.

Regarding prospective works on debris removal applications, novel docking or capture facilities are under study [Bonnal 13a]: harpoon or hook for the ESA ROGER vehicle 1.6(a), capture by a net ((Figure 1.6(b))) or robotic arm for the OTV-2 study led by Astrium and Thales Alenia Space, foam gluing is investigated by the University of Roma and a claw would be the capture tool for the EPFL Clean Space One Program (Figure 1.6(c)). For the DEOS project, led by the German DLR, the system would rely on a more classical robotic arm for berthing [Rank 11, Mühlbauer 12]((Figure 1.3(a))).



(a) (b) (c)

Figure 1.6 – Docking and capture facilities for debris removal applications: hook for the ESA ROGER vehicle (a), net for OTV-2 vehicle (b), claw for the Clean Space One program (c), robotic arm for the DLR DEOS project (d) (Artistic views).

1.2 Navigation measurements and requirements for the far and close range phases

The purpose of this thesis lies in the field of close range navigation solutions for a rendezvous mission, during the final phase of the approach, when relative navigation between the chaser and the target is required, since absolute navigation is not accurate enough or inappropriate for uncooperative targets.

The navigation module of the GNC system involved on the chaser spacecraft requires specific relative measurements to provide its relative localization with respect to the target. Let us review some of these measurements typically used and processed for the close or far range phases of rendezvous missions.

- **Distance** or **range** between the chaser and the target
  It is for instance obtained through triangulation between measurements such as visual measurement and the target known dimensions, or through time of flight or phase shift from a transmitter to a receiver located on the chaser of electro-magnetic wave signal.

- **Line-of-sight** (LOS) direction of the target with respect to the chaser.
  It can be acquired using visual measurement with the position of the target in the image of a calibrated camera or other vision sensors.

- **Relative attitude**
  It corresponds to the relative angular measurements, classically represented by three angles: yaw, pitch and roll.

Range and LOS together correspond to the 3D relative position between the chaser and the target.

Some critical requirements for these measurements are faced at docking or capture steps and are reported on table 1.1. They impose even more stringent requirements during the whole final approach of the rendezvous. Measurements errors have indeed serious

impacts on trajectory deviations. When open loop maneuvers are performed, such as impulsive transfer maneuvers, the initial accuracy of position measurements should be 0.1% of the range. Otherwise, some intermediate mid-course corrections are necessary.

Open loop maneuvers are usually restricted to the far and medium range approaches since potential trajectory deviations at the end of the maneuver could lead to collisions with the target for shorter ranges (less than one or two kilometers). The dispersion for these maneuvers can be reduced by employing closed loop control, and position and velocity accuracy becomes less critical (1% of the range for position measurements are sufficient).

As seen above, the final approach at close range is performed by straight lines, potentially with a fly-around phase, maneuvers. Closed loop control is also carried out and since control errors are involved, position and attitude measurements errors should be a factor of 2 up to 5 lower than the accuracy at capture specified on table 1.1 and velocity measurements errors (through generally differentiation of position measurements) should be a factor 2 lower than the ones at capture, thus imposing a bandwidth in the order of 1Hz for measurement acquisition.

Parameters	**Docking maneuver**	**Berthing maneuver**
Relat. long. velocity	<0.1 m/s	<0.02m/s
Relat. chaser lat. CoM vel.	<0.02m/s	<0.005m/s
Angular rate	<0.1 deg/s for all axes	<0.02 deg/s for all axes
Ang. misalignement	<1.0-5.0 deg for all axes	<10.0 deg for all axes
Lat. misalign. of dock. units	<0.10m	0.10 - 0.50m
Long. misalign.	-	0.10 - 0.50m

Table 1.1 – Requirements at contact [Fehse 08, Astrium 10] in terms of positions, orientations, velocities and angular rates. CoM stands for Center of Mass.

1.3 Navigation sensors for the far and close range phases

With the aim of acquiring measurements and fulfilling the subsequent requirements, accurate and reliable navigation sensors are used. Hereafter are presented some typical operational sensors embedded on the chaser vehicle to provide these measurements or directly the 3D relative position and attitude, essentially in the case of cooperative rendezvous.

1.3.1 Radio Frequency (RF) sensors

The general concept consists in a RF electro-magnetic wave signal transmitted from antennas on the chaser pointing towards the target, reflected on the target with a reflector, and received using an antenna receiver on the chaser. The time of flight of the signal or the phase shift between transmission and reception can provide range measurements, the signal being either pulse modulated for time of flight measurement or continuous for phase shift measurements. It can also provide range-rate measurement through measuring

the Doppler shift of the transmitted frequency when arriving at the receiver.

However, the range of this cooperative system is limited due to the fact that the received reflected signal presents a high signal-to-noise ratio for long ranges and saturation can be observed on amplifiers for short ranges. For this reason, a transponder can be fixed on the target to increase the power of the received signal and to retransmit it to the chaser with an antenna. For LOS measurements, it can be obtained by difference of phase delay or of time of flight delay of the signal reflecting on the target between two antennas located on the chaser. Relative attitude for pitch and yaw can be measured using four antenna beams with different frequencies.

An example of RF sensor is the Russian *Kurs* system, which is one of the first autonomous on-orbit navigation operational system. The system, which is still operating on board *Soyuz* and *Progress* spacecrafts for their rendezvous with the ISS, provides range, range-rate, relative pitch and yaw, from a few hundred kilometers down to contact, with the use of a transponder located on the target. As another example, a RF sensor has more recently been implemented on the Tango/Mango satellites for the PRISMA mission, with the Formation Flying Radio Frequency (FFRF) [Grelier 10]. However, RF sensors are only suitable for cooperative targets and the power consumption and mass of such a system makes it less privileged nowadays.

1.3.2 Relative GPS

Another solution relies on navigation satellite systems such as Global Positioning System (GPS), developed and operated by the US, and Global Orbiting Navigation Satellite System (GLONASS), developed by Russia, or the future Galileo system, developed by the European Union. For relative GPS, the idea is to synchronize and subtract the raw GPS measurements of the chaser and the target using at least four of common navigation satellites. This raw data is then processed into a Kalman navigation filter. But the accuracy of a relative GPS for the considered application is in the order of $1m$ for position and $0.05m/s$ for velocity. Despite it shows much better precision than a simple difference of absolute GPS measurements (accuracy in the order of around 100m for position), it can only be suited for far range approaches.

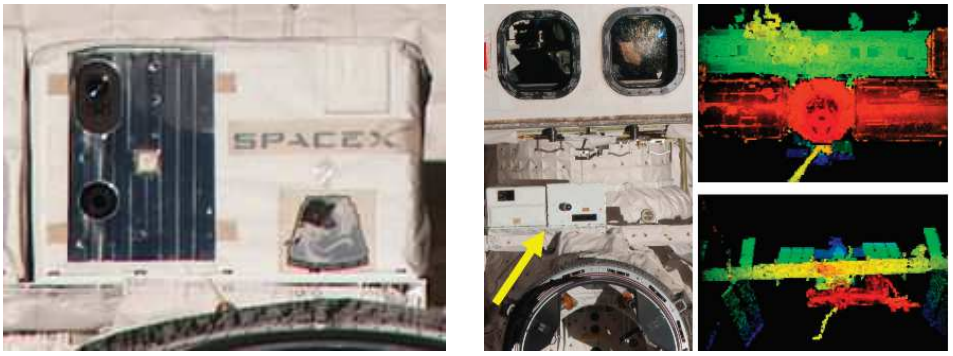
Relative GPS has already been first experimented for autonomous rendezvous systems on the Japanese NASDA ETS-VII [Inaba 00] rendezvous demonstrators for its far range phase. The success of this mission has made it implemented on the HTV resupply vehicle [Kawano 01]. The European ATV vehicles also use Relative GPS for its far range rendezvous phase [Pinard 07]. By definition, relative GPS only concerns cooperative targets.

1.3.3 Range sensors

Regarding uncooperative solutions, which are required for debris removal applications for instance, range sensors are particularly studied. Within range sensors lie various sensors such as Laser Range Finders and LIght Detection and Ranging (LIDAR) [Christian 13]. LIDAR are active sensors consisting in emitting light from the chaser, principally laser light. Reflected on the target, the time of flight to sense it back on the chaser serves to

compute the distance or range with various points on the target, providing a 3D point cloud. Two main types of LIDARs can be distinguished: flash LIDAR, which consist in emitting, within a Field of View, light pulses, and scanning LIDARs, for which a laser beam scans the target, through pan or tilt rotations. The famous DragonEye flash LIDAR (Figure 1.7(a)), developed by Advance Scientific Concepts and SpaceX, was successfully demonstrated on two Space Shuttle Missions (STS-127 on Endeavour, STS-133 on Discovery) [Christian 13, Poberezhskiy 12], and is currently operating on the SpaceX Dragon spacecraft for its close range approach (200m-10m) with the ISS, until berthing and getting grabbed by the ISS Canadarm robotic arm.

As promising demonstrators let us mention the systems proposed by Neptec, the Laser Camera System (LCS) [Samson 04] and more recently the TriDAR system [Ruel 05] (Figure 1.7(b)), a 3D scanner which combines laser triangulation with time of flight ranging (LIDAR). Both systems aims at estimating the relative pose by processing 3D range data of the target, using for instance for TriDAR an Iterative Closest Point algorithm to match the resulting 3D point cloud with the known 3D model of the target (here the ISS). The ability of the LCS was verified during the STS-105 mission in 2001 and on-board Shuttle Discovery during STS-128 in 2009 and STS-135 in 2011 for TriDAR [Ruel 08, Ruel 10], which has been selected as the navigation sensor for the Orbital Cygnus ISS resupply vehicle, whose first flight was carried out very recently on September 18^{th} 2013.Using range sensors has also been considered in [Lichter 04], estimating the state, shape, and model parameters of space objects from range images acquired by a team of cooperative sensors. We can finally notice the works of [deLafontaine 06], for planetary landing purposes, which proposed to use a LIDAR sensor to accomplish navigation with respect to the target by matching constellation of 3D landmaks.



(a) (b)

Figure 1.7 – The DragonEye LIDAR sensor on Space Shuttle Discovery for STS-133 mission (a). (b) features the TriDar sensor mounted on Space Shuttle Discovery for STS-128 mission (left) and a TriDar image of the ISS (right) [Ruel 08].

1.3.4 Camera sensors and computer vision-based navigation for the final approach and proximity operations

Another famous class of sensors regards computer vision and would be particularly suitable for the close range approach in autonomous rendezvous missions, especially for uncooperative targets, since they require no data from the target. Besides vision sensors

require few hardware facilities and low power consumption. The next section reviews the different vision-based navigation systems which have been studied, experimented or implemented on operational systems.

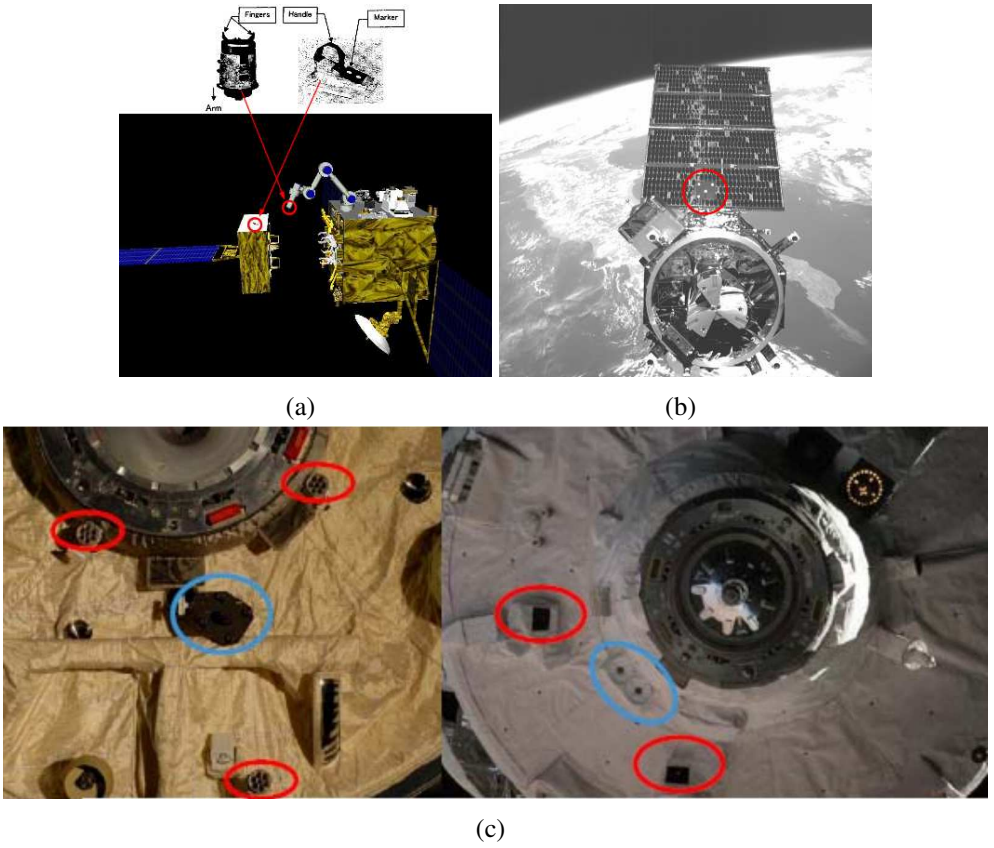
The ETS-VII demonstrator [Inaba 00], conducted by the Japanese NASDA in 1998-1999, performed successful rendezvous, capture and docking operations between two satellites with the chaser equipped with a robotic arm. The capture was achieved using computer vision, with a monocular camera fixed on the robotic arm and fiducial markers located on the target, to measure the relative position and orientation parameters, or pose (a 3 points marker for full pose measurement, 2 points marker for full pose except yaw, during final robotic grasping operation, Figure 1.8(a)), making this system the first vision-based autonomous rendezvous demonstrator, operating at close range.

Another famous vision-based rendezvous and docking demonstrator is the Advanced Video Guidance Sensor used in the DARPA Orbital Express mission in 2007 [Howard 08, Friend 08], which completed various tasks such as autonomous capture with a robotic arm fixed on ASTRO, refueling of NextSat, formation flying. For full relative pose estimation, laser diodes mounted on the chaser (ASTRO) were used to illuminate corner-cube retro-reflective markers on the target (Nextsat, Figure 1.8(b)). A similar technology was also implemented on the ATV [Blarre 04, Pinard 07, Strandmoe 08]. At close range, pulse laser beams reflect on an a set of 26 retro-reflectors (with five of them lying a corner-cube retro-reflector) installed on the ISS-Zvezda target module (Figure 1.8(c)). Reflections are then imaged by a camera set up on the ATV, the resulting image being processed to provide range and azimuth/elevation of the target from 250 meters, plus relative attitude at closer range (from 30m). These measurements are fused with measurements provided by a telegoniometer.

With a monocular camera, some prospective studies [Blais 10, Center 04, Woffinden 07, Tweddle 10] also propose to rely on easy to detect and track known patterns or fiducial markers installed on the target.

However, all these systems based on vision sensor are cooperative.

Few studies address this issue of vision-based uncooperative navigation. The German Space Center has led the (DLR) Deutsche Orbitale Servicing Mission (DEOS) [Rank 11, Mühlbauer 12] project, which intends to accomplish capture, berthing, stabilization and docking operations with a tumbling uncooperative target satellite with a robotic arm. An on-orbit demonstrator is planned to be launched from 2018. Here a vision-based pose estimation solution relying on stereo cameras is proposed [Rank 11]. Using stereo has also been considered in [Jasiobedzki 01, Dionnet 07, Oumer 12a]. With a monocular camera, a local feature matching approach had been selected in [Tingdahl 10]. It consists in the extraction of invariant features in the image that are matched to a database built from preliminary learning sessions, but has shown to be computationally prohibitive and sensitive to distance, illumination, relative orientation, and occlusions of the target. Also, the Orbital Life Extension Vehicle (OLEV), whose activities are currently on hold, is a program led by Kayser and Sener to provide servicing operations to geostationary satellites and provide debris removal abilities [Miravet 08]. For the rendezvous approach and docking phases navigation system, [Miravet 08] presents a monocular edge-based solu-



(a) (b)

(c)

Figure 1.8 – Vision-based navigation solutions onboard rendezvous systems. ETS-VII mission (a): robotic arm end-effector and the 2 points marker installed on the target [Inaba 00]. Corner-cube markers on NextSat for the Orbital Express mission (b).On (c) are depicted the retro-reflectors installed on the ISS-Zvezda module (left, in blue the corner-cube retro-retroreflector, in red the ones processed by both the vision system and the telegoniometer) and the CCD camera located on the ATV (right, in blue) [Strandmoe 08].

tions to detect and track the target based, but limited to a single viewpoint since a simple 2D geometrical model of the target is processed. [Kelsey 06] computes the relative pose based on the knowledge of the 3D CAD model of the target through an edge-based tracking process. In order to tackle the problem without the use of fiducial markers or a priori knowledge of the target shape, [Augenstein 11, Oumer 12b] proposed monocular SLAM techniques.

In the field of landing on planets or on small solar systems bodies, a widely known example is the landing of Hayabusa probe on Itokawa asteroid in 2005 [Kubota 06, Yano 06], on tracking both visual natural (extracted on the asteroid surface) and fiducial (target marker dropped on the asteroid) landmarks, along with LIDAR (for far range) and Laser rangefinder (for close range) sensors, despite difficulties to achieve the touchdowns. Some other works have studied pin point landing based on visual landmarks, matched with an off-line database [VanPham 12, Delaune 12].

1.4 Scope of the study and testing facilities

In the field of space rendezvous and proximity operations, an objective of this thesis has been to design a relative navigation solution during the close range phase of a rendezvous mission between a chaser and an uncooperative target, for on-orbit servicing or space debris removal applications. The starting secure distance for this autonomous terminal phase would be set around 1000m.

For several reasons, a vision-based solution using a monocular camera sensor, is proposed. With respect to other sensors (vision or other), a monocular camera is indeed a cheap and mature system, with a small form factor and a low power consumption, and it can be easily calibrated and installed on the chaser vehicle. In contrast to range data sensor such as LIDAR, it can operate on further ranges, depending on the camera Field of View (FoV).

The presented solution is also based on natural features of the target since it should suited for uncooperative targets, with the a priori knowledge of the (complete or partial) 3D CAD model of the target. Our applications indeed deals with industrial objects such as satellites for which their 3D models can be provided.

In order to test and validate the methods which have elaborated for this thesis, several testing procedures have been proposed.

1.4.1 Qualitative tests on various real image data

Dealing with aerospace and space applications faces issues when it comes to experimentally validating solutions and processing real data in this context. It is particularly the case for computer vision solutions since obtaining exploitable image sequences involving a rendezvous approach is not an easy task: rendezvous missions are not achieved on a daily basis and remain exceptional operations, camera sensors currently mounted on spacecraft facilities are usually not installed for visual measurement purposes, and operators (NASA, ESA, JAXA,...) can be reluctant to provide such a critical data, as well as the corresponding ground truth provided by other sensors.

However, a few available image sequences concerning recent rendezvous operations can be found on the Internet on Youtube or on the NASA video repository. Although associated ground truth is not available, as well as the camera parameters, they can be decently processed for qualitative evaluation. Some examples of such sequences can be seen on Figure 1.9, showing rendezvous maneuvers of the Space shuttle[5] and Soyuz spacecraft[6] towards the ISS.

Our methods are based on the knowledge of the 3D CAD model of the target, which can be a constraining requirement. Nevertheless, the aimed applications deal with industrial objects for which complete 3D CAD model can be provided or easily found on the Internet on platforms such as Google 3D Warehouse[7] or on Space simulators online li-

[5] http://youtu.be/ZYb0p991x1Y
[6] http://youtu.be/MlRmTgsDYjk
[7] http://sketchup.google.com/3dwarehouse/

brary such as Celestia[8]. Astrium has also provided us with 3D models of satellites. Some examples of such models can be seen on Figure 4.2.

(a) (b)

Figure 1.9 – Examples of image sequences: pitch maneuver of the Atlantis Space Shuttle (STS-135) (a) and Soyuz spacecraft undocking from the ISS (b).

(a) (b) (c)

Figure 1.10 – Examples of 3D models directly used in our experiments. (a) is the Soyuz TMA spacecraft, (b) is the Atlantis Space Shuttle and (c) a Spot satellite.

1.4.2 Quantitative tests on synthetic images

The available real data can only give us a visual qualitative evaluation. With the aim of comparing our vision-based navigation contributions with respects to state-of-the-art methods and measuring the compliance of these solutions to the requirements presented in section 1.2, we have also carried out quantitative tests on some synthetic data provided with ground truth. For this purpose, Astrium has provided us with a realistic image simulator for space environments. This library, referred as Surrender, is based on a ray-tracing image rendering engine overlying OpenSceneGraph rendering engine and OpenGL graphics library. It enables to easily build 3D scenes described by a set of planets, satellites and light sources. The engine expects a list of real planets and a star named "sun" which will be used to set the sun position and size (as a light source):

[8] `http://www.shatters.net/celestia/`

- **Light sources**
  All simulated lights have quadratic attenuation. The sun is simulated as a point light but is locally considered a directionnal light when rendering shadow maps (if rendering satellites through OpenGL). All other light sources are spot lights. They have a position, a direction, a color, a cutoff angle and an exponent which controls how light power decreases when reaching the cutoff angle.

- **Planets**
  Planets have a name, a radius, a position, an attitude, a shader and a set of textures. They are represented as discretized spheres to be rendered through OpenGL but as real spheres for the raytracer (only center and radius).

- **Satellites**
  Satellites are 3D models loaded from a file (3DS, VRML, ...). All textures loaded with a model are interpreted as diffuse textures, and a specular, an emission and a normal texture can be added.

This simulator is also enable to simulate realistic orbital trajectories for both the chaser and the target.

1.4.3 Test bed using a robotic platform and a satellite mock-up

To implement our algorithms on a more realistic vision-based rendezvous context, Astrium provided a complete 3D-model and a reduced (1/50) mock-up of Amazonas-2. It is a telecom satellite builtfrom the Eurostar-3000 platform, and similar to the one used for HARVD [Astrium 10, Tingdahl 10]. Amazonas-2 was launched in 2009 for Spanish company Hipsasat to cover the American (especially South America) position. It is located on a Geostationary Orbit. To simulate an approach with this mock-up, we have used the Afma6 robotic arm available in the Inria robotic platform. This 6 Degrees of Freedom (DoF) robotic arm, with a camera mounted on its end-effector, enables to have regular and realistic movements . Sun illumination can also be simulated by a spot light located around the scene. A typical experimental set-up can be seen on Figure 1.11. This set-up has been used to carry out both open loop and closed-loop tests, using visual servoing.

1.5 Conclusion

The goal of this chapter was to present the issues related to autonomous space rendezvous and proximity operations, setting the scope of this thesis. The concept of space rendezvous has been introduced, including some historical aspects, some operational concerns and some challenges which have nowadays to be faced.

Incorporating autonomy for space rendezvous operations is indeed currently actively questioned, since two main crucial problems are to be handled. The first problem regards on-orbit servicing of satellite or space facilities such as the International Space Station. The second problem concerns the active removal of space debris whose population has critically expanded for the last years and needs to be mitigated.

For these reasons and since the autonomy of such systems highly depends on navigation issues, much attention has been paid on designing autonomous navigation solutions,

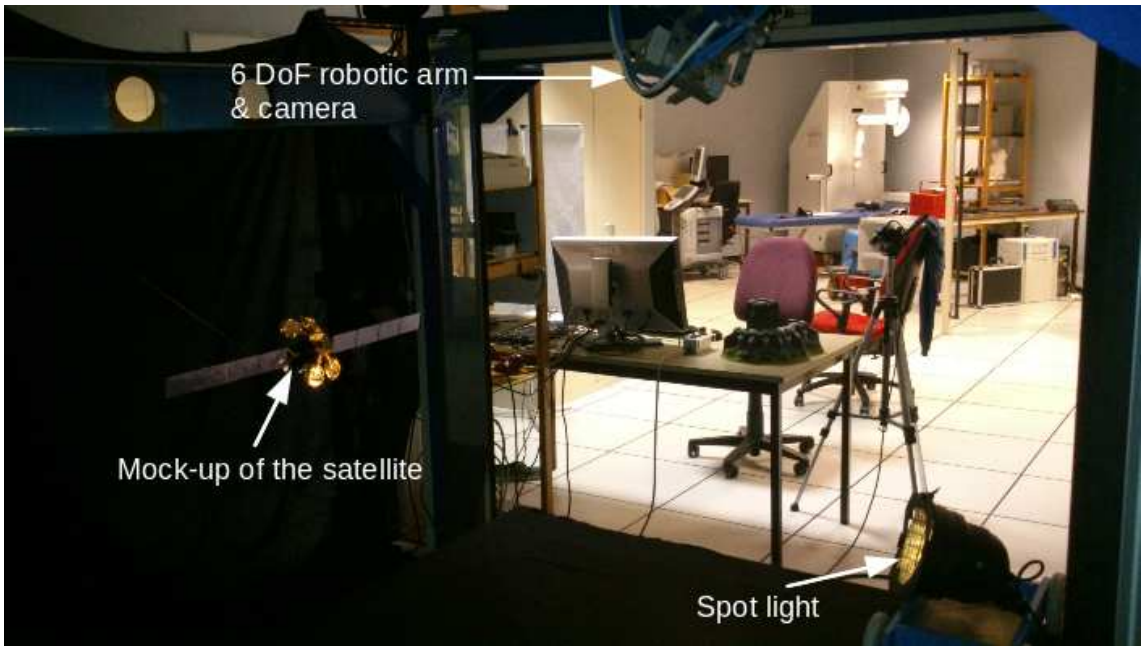


Figure 1.11 – Experimental set-up on the Lagadic robotic platform.

especially for the perilous case of the final approach between the chaser and target spacecrafts, which impose stringent measurement requirements. In this sense, computer vision solutions are currently widely studied. For operational or demonstration concerns, most of the implemented techniques rely on markers installed on the target, making them exclusively suitable for cooperative operations.

Handling the problem of uncooperative vision-based solution, prospective studies propose the use of active range sensors and passive monocular or stereo cameras. Monocular cameras appear to be a flexible, low-cost and convenient sensor, but few works have been proposed based on this sensor. Let us however mention some monocular SLAM-based methods, but which do not provide an absolute localization with respect to the target.

The overall goal of this thesis is thus to design a monocular vision-based localization solution between the chaser and the known target, suited for uncooperative rendezvous missions, and possibly handling both issues of on-orbit servicing and space debris removal. More precisely, the use of 3D CAD model of the target is considered, so that absolute localization can be performed.

Different tools and facilities have been made available to us and have been described in this chapter.

The next chapter tackles general issues related to monocular computer vision and more precisely related to the two major challenges of this thesis which are initial visual localization of the target, through pose estimation by detection, and visual localization through tracking.

Chapter 2

Background on computer vision

In computer vision, the problem of 3D object localization or tracking is closely related to the estimation of geometrical transformations. This chapter first introduces some mathematical background and tools to formalize these transformations. For this purpose, Euclidean geometry allows to describe the way an object is moving with respect to a scene or another object in the 3D space, through 3D rigid transformations (section 2.1). By considering the camera projection model and modeling digital images provided by the camera, we present how the object is then projected and perceived in the image and how 2D image plane transformations can be handled (section 2.2). These 2D transformations can be directly be used for the purpose of 2D tracking. With 3D tracking, they can be indirectly used to retrieve the relative state between the object and the camera. Section 2.3 thus presents how the 3D rigid transformation or pose between the camera and the target object can be estimated based on information extracted from an image or a sequence of images. A review of existing concepts, methods and techniques is therefore proposed, regarding two aspects of the vision-based pose estimation problem. The first aspect concerns pose estimation by detection, given an initial input image (section 2.4). It is a required step to initialize the second aspect, that is pose estimation by frame-by-frame tracking, given a sequence of successive input images (section 2.5).

2.1 Euclidean geometry and 3D rigid transformation

Describing the relative state between a scene or an object and a camera in space can be done using 3D rigid transformations between their respective attached coordinate frames. The next section introduces how a 3D rigid transformation between two frames can be represented.

2.1.1 Coordinate frames and homogeneous matrix

Let us define two coordinate frames $\mathcal{F}_a$ and $\mathcal{F}_b$ in the 3D Euclidean space $\mathbb{E}^3$, which is modeled by the real coordinate space $\mathbb{R}^3$.

Both frames are defined by their origins $\mathcal{O}_a$ and $\mathcal{O}_b$ and their sets of three orthonormal axes $(\mathbf{x}_a, \mathbf{y}_a, \mathbf{z}_a)$ and $(\mathbf{x}_b, \mathbf{y}_b, \mathbf{z}_b)$. A 3D geometric rigid transformation allows to express the coordinates ${}^b\mathbf{X} = \begin{bmatrix} {}^bX & {}^bY & {}^bZ \end{bmatrix}^T$ of a 3D point $\mathcal{X} \in \mathbb{R}^3$ in $\mathcal{F}_b$, based on its coordinates ${}^a\mathbf{X} = \begin{bmatrix} {}^aX & {}^aY & {}^aZ \end{bmatrix}^T$ in $\mathcal{F}_a$ (see Figure 2.1). This transformation in $\mathbb{R}^3 \times SO(3)$, from $\mathcal{F}_a$ to $\mathcal{F}_b$, consists in a position transformation, modeled by a translation vector ${}^b\mathbf{t}_a$, and in an rotation transformation, modeled by a rotation matrix ${}^b\mathbf{R}_a$, so that:

$$ {}^b\mathbf{X} = {}^b\mathbf{R}_a \, {}^a\mathbf{X} + {}^b\mathbf{t}_a \tag{2.1} $$

where ${}^b\mathbf{t}_a$ is a 3×1 vector and ${}^b\mathbf{R}_a$ a 3×3 matrix. ${}^b\mathbf{R}_a \in SO(3)$. $SO(3)$ the Special Orthogonal group, which a rotation matrix belongs. It is defined by:

$$ SO(3) = \left\{ \mathbf{R} \in \mathbb{R}^{3\times3} \mid \mathbf{R}^\top \mathbf{R} = \mathbf{I}_3, det(\mathbf{R}) = 1 \right\}. \tag{2.2} $$

Projective space and homogeneous coordinates

Relation (2.1) is affine and using the projective space allows to write it linearly. Indeed, a point $\mathcal{X}$ in $\mathbb{E}^3$, defined by its cartesian coordinates $\mathbf{X}$ in $\mathbb{R}^3$, can be equivalently represented by a point $\overline{\mathcal{X}}$ in the projective space $\mathbb{P}^3$, described by its homogeneous coordinates $\overline{\mathbf{X}} = \begin{bmatrix} \lambda\mathbf{X}^T & \lambda \end{bmatrix}^T = \begin{bmatrix} \mathbf{X}^T & 1 \end{bmatrix}^T$, with λ a scalar. $\mathbb{P}^3$ can be seen as an extension of $\mathbb{E}^3$ such that all $\lambda\mathbf{X} \in \mathbb{R}^3$ correspond to a unique point $\overline{\mathcal{X}}$.

As a consequence, using the normalized homogeneous coordinates ${}^a\overline{\mathbf{X}} = \begin{bmatrix} {}^a\mathbf{X}^T & 1 \end{bmatrix}^T$ and ${}^b\overline{\mathbf{X}} = \begin{bmatrix} {}^b\mathbf{X}^T & 1 \end{bmatrix}^T$ of $\mathcal{X}$ respectively in $\mathcal{F}_a$ and $\mathcal{F}_b$, equation (2.1) can be rewritten as:

$$ {}^b\overline{\mathbf{X}} = {}^b\mathbf{M}_a \, {}^a\mathbf{X} \qquad \text{with:} \qquad {}^b\mathbf{M}_a = \begin{bmatrix} {}^b\mathbf{R}_a & {}^b\mathbf{t}_a \\ \mathbf{0} & 1 \end{bmatrix}. \tag{2.3} $$

${}^b\mathbf{M}_a$ is defined as the homogeneous matrix describing the 3D rigid transformation in $\mathbb{P}^3$ from $\mathcal{F}_a$ to $\mathcal{F}_b$, in the Special Euclidean group $SE(3)$, which is defined by:

$$ SE(3) = \left\{ \mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times3} & 1 \end{bmatrix} \mid \mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^3 \right\}. $$

2.1.2 Parametrization of the rotation

For numerical concerns, the rigid transformation ${}^b\mathbf{M}_a$ needs to be properly parametrized. While using translation parameters of ${}^b\mathbf{M}_a$ is natural, it would be awkward to directly use the 3×3 rotation matrix in ${}^b\mathbf{M}_a$ and its nine non-independent parameters. Indeed, some non-linear constraints have to be faced and added. As specified by equation (2.2), a rotation matrix $\mathbf{R}$ requires to keep its three columns to be unit vectors and to be orthogonal. In order to respect these constraints, other convenient parametrization solutions are possible.

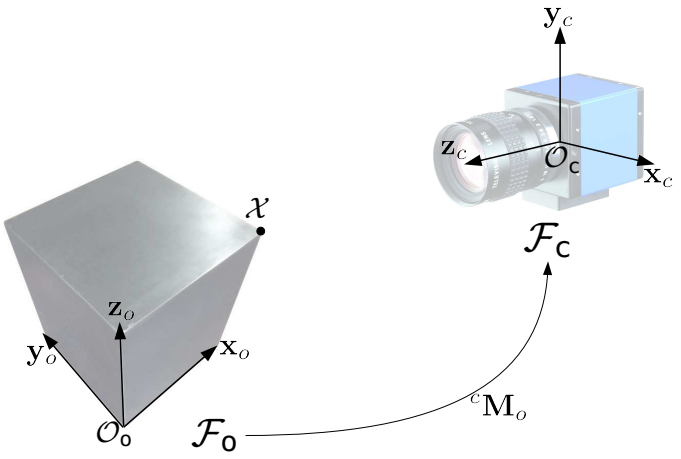


Figure 2.1 – Representation of the 3D space using Euclidean geometry, in the case of an object o and camera c.

2.1.2.1 Euler angles

One popular and simple way is to use Euler angles. Since a rotation matrix can be defined as a composition of three rotations around the three orthonormal axes $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ of the space in $\mathbb{R}^3$, it can be written as the product of three 1D rotation matrices around each axis, with corresponding angles r_x, r_y and r_z. An important convention concerns the order of the 1D rotations, so that a rotation can parametrized by the vector (r_x, r_y, r_z), or for instance (r_z, r_y, r_x). While getting a rotation matrix from Euler angles is simple, the inverse operation is also easy, by identifying the coefficients of the rotation matrix with their analytical expression, the solution being non-unique. For these reasons Euler angles have been widely used, especially in aeronautics, for their convenience and their intuitive meaning. But they suffer from one major drawback, known as *gimbal lock*: if two rotation axes are aligned then one degree of freedom is lost.

2.1.2.2 Quaternions

Quaternions are another common representation for rotations in 3D space. They are hypercomplex numbers, which can be written as a scalar plus a 3D vector. A rotation by an angle θ about a unit vector $\mathbf{e}$ can then be represented by the quaternion:

$$\mathbf{q} = \begin{bmatrix} \cos(\frac{\theta}{2}) & \mathbf{e}^T \sin(\frac{\theta}{2}) \end{bmatrix}^T . \tag{2.4}$$

With this parametrization *gimbal lock* can be avoided but it obeys the constraint to have a norm equal to one, resulting in an increased algorithmic complexity when using numerical optimization techniques.

2.1.2.3 Axis and angle of rotation

Any rotation can be described by a vector $\theta\mathbf{u} = \begin{bmatrix} \theta u_x & \theta u_y & \theta u_z \end{bmatrix}^T$, with $\theta = \|\theta\mathbf{u}\|$ the angle of the rotation, about an axis of direction $\mathbf{u}$, with $\|\mathbf{u}\| = 1$. $\theta\mathbf{u}$, known as the

exponential canonical representation, can then be linked to the rotation matrix through its exponential map (see [Ma 04] for an exhaustive presentation of rigid transformations):

$$\mathbf{R} = exp([\theta\mathbf{u}]_\times) = \sum_{n=0}^{\infty} \frac{[\theta\mathbf{u}]_\times^n}{n!}. \tag{2.5}$$

For a vector $\mathbf{v} = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^T$, $[\mathbf{v}]_\times$ denotes the associated skew symmetric matrix:

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & -v_x \\ -v_y & v_x & 0 \end{bmatrix}. \tag{2.6}$$

$\mathbf{R}$ can also be evaluated using Rodrigues' formula:

$$\mathbf{R} = \mathbf{I}_3 + \frac{\sin\theta}{\theta}\,[\mathbf{u}]_\times + \frac{(1-\cos\theta)}{\theta^2}\,[\mathbf{u}]_\times^2. \tag{2.7}$$

The inverse operation giving $\theta\mathbf{u}$ from $\mathbf{R}$ is done using the following equations:

$$\cos\theta = \frac{1}{2}\,(\,\text{trace}(\mathbf{R}) - 1) \tag{2.8}$$

and

$$\sin\theta\;[\mathbf{u}]_\times = \frac{1}{2}(\mathbf{R} - \mathbf{R}^\top). \tag{2.9}$$

Setting $\theta > 0$, $\sin\theta \neq 0$, $\theta\mathbf{u}$ can be uniquely determined using (2.8) and (2.9):

$$\theta = acos\left[\frac{\mathbf{R}_{11} + \mathbf{R}_{22} + \mathbf{R}_{33} - 1}{2}\right] \tag{2.10}$$

$$\theta\mathbf{u} = \frac{1}{2\,\frac{\sin(\theta)}{\theta}} \begin{bmatrix} \mathbf{R}_{32} - \mathbf{R}_{23} \\ \mathbf{R}_{13} - \mathbf{R}_{31} \\ \mathbf{R}_{21} - \mathbf{R}_{12} \end{bmatrix} \tag{2.11}$$

The singularity occurring when θ is close to zero can be avoided by replacing $\frac{\sin(\theta)}{\theta}$ and $\frac{(1-\cos\theta)}{\theta^2}$ by the first two terms of their Taylor expansions. However singularities are still encountered when $\theta = n\pi$ with $n \geq 1$. Nevertheless they can be managed by setting $\theta\mathbf{u}$ to a new equivalent rotation far from the singularity that is approached.

With the exponential map, a rotation can be represented by three parameters, the *gimbal lock* is avoided and no additional constraint are to be dealt with. For these reasons, this representation is suitable with numerical optimization issues and will be considered in the reminder of this thesis.

2.2 Image formation

In the previous section has been exposed the way a 3D rigid transformation can be modeled. This representation can be applied to the case of computer vision, considering a scene and a camera in space. Since the camera provides 2D images of the scene, this section thus aims at describing how the scene projects itself in the image, how this 3D-2D transformation can be modeled and how 2D transformations in the image plane can be represented.

2.2.1 Pinhole camera model

In order to model the projection in the image provided by the camera, we use the pinhole camera model. It is based on a perspective projection [Faugeras 93, Hartley 01], which is commonly used in computer vision, since it approximates well the physical model of a real camera and enables a convenient and linear expression of the 3D-2D transformation between the scene and the image, using homogeneous coordinates. This model relies on the concept that all light rays emitted or reflected by the scene and collected by the camera converge towards one point, the optical center, or projection center. This point is set to be the origin of the camera frame $\mathcal{F}_c = (\mathcal{O}_c, \mathbf{x}_c, \mathbf{y}_c, \mathbf{z}_c)$. The image plane $\mathcal{P}_\pi$, or projection plane, on which light rays are projected, is parallel to the plane $(\mathbf{x}_c, \mathbf{y}_c)$ and is located at a distance f on the $\mathbf{z}_c$ axis from the optical center. f is called the focal distance and the projection of $\mathbf{O}_c$ on $\mathcal{P}_\pi$ is known as the principal point. Let $\mathcal{X}$ be a point in the 3D space represented by its coordinates expressed in the camera frame ${}^c\mathbf{X} = \begin{bmatrix} {}^cX & {}^cY & {}^cZ \end{bmatrix}^T$. The 2D coordinates $\mathbf{x} = \begin{bmatrix} x & y \end{bmatrix}^T \in \mathbb{R}^2$ of the projection of $\mathcal{X}$ in the image plane are given by (see Figure 2.2):

$$\begin{bmatrix} x \\ y \end{bmatrix} = \frac{f}{{}^cZ} \begin{bmatrix} {}^cX \\ {}^cY \end{bmatrix} \tag{2.12}$$

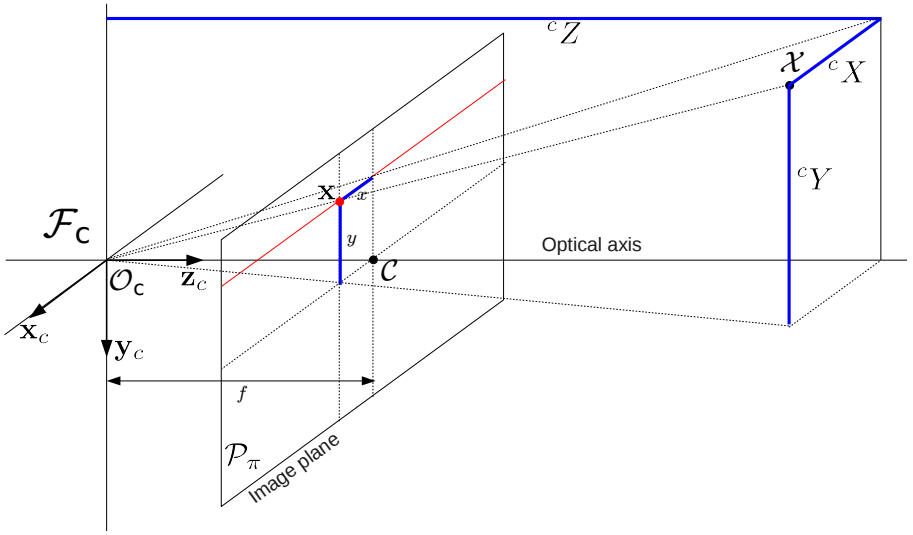


Figure 2.2 – Pinhole camera projection model.

Using the corresponding homogeneous coordinates ${}^c\overline{\mathbf{x}} = \begin{bmatrix} x & y & 1 \end{bmatrix}^T$ and ${}^c\overline{\mathbf{X}} = \begin{bmatrix} {}^cX & {}^cY & {}^cZ & 1 \end{bmatrix}^T$, equation (2.12) can be linearly rewritten, using a projective transformation $\mathbf{A}$ from $\mathbb{P}^3$ to $\mathbb{P}^2$:

$${}^c\overline{\mathbf{x}} = \mathbf{A}\,{}^c\overline{\mathbf{X}} \qquad \text{with} \qquad \mathbf{A} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.13}$$

These normalized coordinates are expressed in the metric space and need to be converted into pixel coordinates in the digital image corresponding to the image plane.

2.2.2 Digital images

A digital image results from a discretization of the image plane into a regular grid whose rectangular elements are called pixels. A pixel is defined by its position on the grid and its color, which is encoded by the intensity perceived by the corresponding electronic image sensor (such as CCD or CMOS sensors) placed on the image plane, on the three color channels (red, green and blue). For a point $\mathbf{x}$, the transformation from its normalized metric coordinates $\mathbf{x} = \begin{bmatrix} x & y \end{bmatrix}^T$ to its pixel coordinates $\mathbf{p} = \begin{bmatrix} u & v \end{bmatrix}^T \in \mathbb{R}^2$ is parametrized by four degrees of freedom, which are the coordinates (u_0, v_0) of the principal point in pixels and the pixel sizes (l_x, l_y) in meters (see Figure 2.3). This transformation can then be written as:

$$\begin{cases} u &= u_0 + \frac{1}{l_x} x \\ v &= v_0 + \frac{1}{l_y} y \end{cases} \tag{2.14}$$

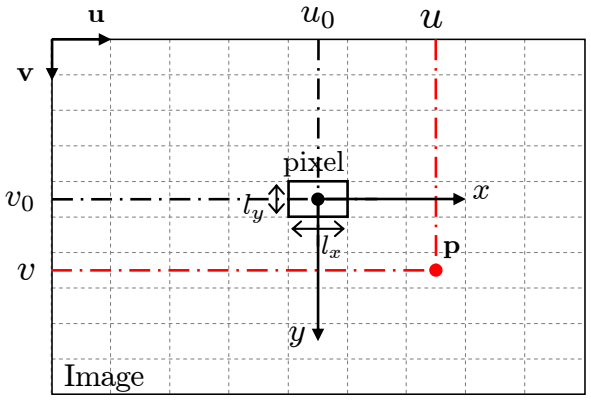


Figure 2.3 – Digital image and pixel coordinates.

Once again, this transformation can be expressed in a linear form using homogeneous coordinates, using the projective transformation $\mathbf{K}'$ from $\mathbb{P}^2$ to $\mathbb{P}^2$:

$$\overline{\mathbf{p}} = \mathbf{K}'\overline{\mathbf{x}} \qquad \text{with} \qquad \mathbf{K}' = \begin{bmatrix} \frac{1}{l_x} & 0 & u_0 \\ 0 & \frac{1}{l_y} & v_0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.15}$$

Equations (2.12) and (2.15) enable to express how a point $\mathcal{X}$ defined in the camera frame $\mathcal{F}_c$ with its homogeneous coordinates ${}^c\overline{\mathbf{X}} = \begin{bmatrix} {}^cX & {}^cY & {}^cZ & 1 \end{bmatrix}^T$ projects itself in the digital image:

$$\overline{\mathbf{p}} = \mathbf{K}\mathbf{\Pi}\overline{\mathbf{X}} \qquad \text{with} \qquad \mathbf{\Pi} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.16}$$

$$\text{and} \qquad \mathbf{K} = \begin{bmatrix} p_x & 0 & u_0 \\ 0 & p_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.17}$$

or equivalently:

$$\overline{\mathbf{p}} = \mathbf{K}\mathbf{X} \tag{2.18}$$

with $p_x = \frac{f}{l_x}$ and $p_y = \frac{f}{l_y}$. The pinhole camera can thus be modeled by $\mathbf{K}$, which is called the intrinsic parameters matrix. If $\mathcal{X}$ is expressed in the object frame $\mathcal{F}_o$ with ${}^o\overline{\mathbf{X}} = \begin{bmatrix} {}^oX & {}^oY & {}^oZ & 1 \end{bmatrix}$, equation (2.18) becomes:

$$\overline{\mathbf{p}} = \mathbf{K}\mathbf{\Pi}\,{}^c\mathbf{M}_o\,{}^o\overline{\mathbf{X}} \tag{2.19}$$

with ${}^c\mathbf{M}_o$ the homogeneous matrix associated to the rigid transformation from $\mathcal{F}_o$ to $\mathcal{F}_c$, defined in section 2.1.1. Its parameters are called the extrinsic parameters and depend on the geometry of the scene, whereas the intrinsic parameters, making of $\mathbf{K}$, are inherent to the considered camera sensor and are usually provided by the constructor. Intrinsic parameters can nevertheless be estimated using calibration techniques [Brown 71, Tsai 86, Marchand 02], which generally consist in solving a set of equations relating some known 3D points ${}^o\overline{\mathbf{X}}$ with known points $\overline{\mathbf{p}}$ in the image. The pinhole camera model suppose a perfect optical system which respects Gauss conditions, so that light rays can be represented by lines. However some deformations or distortions can be observed for some systems, such as cameras with wide FoV. These deformations are mainly due to radial distortions which can be modeled by [Faugeras 93, Hartley 01]:

$$x = x_d(1 + k_1 r^2 + k_2 r^4) \tag{2.20}$$
$$y = y_d(1 + k_1 r^2 + k_2 r^4) \tag{2.21}$$

where (x, y) is a point in the image plane using a pure perspective model and (x_d, y_d) the corresponding point with the distorted model. $r^2 = x_d^2 + y_d^2$, and k_1 and k_2 are the parameters of these distortions and can be estimated, along with (u_0, v_0, p_x, p_y), with specific calibration methods such as [Brown 71, Tsai 87, Marchand 02, Stein 97]. Tangential distortion can also be observed but can usually be neglected with regards to the radial distortions.

2.2.3 Image plane transformations

3D rigid transformations between the camera and the object involve 2D transformations in the image plane. The nature of the relative motion between the camera and the object (translation, rotation) and the nature of the object (planar, non-planar) determines the type of image transformation. In some cases it is possible to define a function $\mathbf{w}$, called a warping function, which maps a point $\mathbf{x}_k = \begin{bmatrix} x_k & y_k \end{bmatrix}^T$ in an image $\mathbf{I}_k$ to a corresponding point $\mathbf{x}_{k+1} = \mathbf{w}(\mathbf{x}_k)$ in $\mathbf{I}_{k+1}$.

Translation

This is the most simple case, for which $\mathbf{x}_k$ is moved to $\mathbf{x}_{k+1}$ by a translation motion. $\mathbf{w}$ is simply parametrized by a 2D translation vector (along $\mathbf{x}$ and $\mathbf{y}$) $\mathbf{t} \in \mathbb{R}^2$, giving $\mathbf{x}_{k+1} = \mathbf{t} + \mathbf{x}_k$, with $\mathbf{t} = \begin{bmatrix} t_x & t_y \end{bmatrix}^T$.

Similarity transformation

It consists in a translation $\mathbf{t}$ combined with a 2D rotation, represented by a rotation matrix $\mathbf{R}$, and a scaling factor $s \in \mathbb{R}^*$: $\mathbf{w}(\mathbf{x}_k) = s\mathbf{R}\mathbf{x}_k + \mathbf{t}$. This transformation models the projection of a planar object that remains parallel to the image plane during the motion. However, it can be applied for a non-planar object when this object is sufficiently far from the camera and can then be approximated by a plane.

Affine transformation

This transformation generalizes the similitude since it can model a stretching of the projection of the object along the $\mathbf{x}$ and $\mathbf{y}$ axis of the image plane. It is composed of a linear transformation represented by a matrix $\mathbf{A} \in \mathbb{R}^{2\times 2}$ and a translation $\mathbf{t}$: $\mathbf{w}(\mathbf{x}_k) = \mathbf{A}\mathbf{x}_k + \mathbf{t}$.

Homography

A homography is a projective transformation from $\mathbb{P}^2$ in $\mathbb{P}^2$. As opposed to the previous transformations, it can model perspective effects in the image for a plane $\mathcal{P}$ in the 3D space. It consists in mapping $\mathcal{P}$ from an image $\mathbf{I}_k$ associated to the camera frame $\mathcal{F}_c^k$ to an image $\mathbf{I}_{k+1}$, corresponding to $\mathcal{F}_c^{k+1}$. Let $\mathbf{n}_k$ denote the normal vector to $\mathcal{P}$ expressed in $\mathcal{F}_c^k$ and d_k the distance between $\mathcal{P}$ and the center of projection of $\mathcal{F}_c^k$. For each point $\mathcal{X}$ belonging to $\mathcal{P}$, its homogeneous coordinates ${}^c\overline{\mathbf{X}}_k$ in $\mathcal{F}_c^k$ verify:

$$\mathbf{n}_k^T \, {}^c\overline{\mathbf{X}}_k = d_k. \tag{2.22}$$

With (2.1) and the rigid transformation between $\mathcal{F}_c^k$ and $\mathcal{F}_c^{k+1}$, parametrized by a rotation matrix ${}^{k+1}\mathbf{R}_k$ and the translation ${}^{k+1}\mathbf{t}_k$, we can express the coordinates ${}^c\mathbf{X}_{k+1}$ of $\mathcal{X}$ in $\mathcal{F}_c^{k+1}$:

$${}^c\mathbf{X}_{k+1} = {}^{k+1}\mathbf{R}_k \, {}^c\mathbf{X}_k + {}^{k+1}\mathbf{t}_k. \tag{2.23}$$

This equation can be factorized using (2.22) into:

$${}^c\mathbf{X}_{k+1} = {}^{k+1}\mathbf{H}_k \, {}^c\mathbf{X}_k \qquad \text{with} \qquad {}^{k+1}\mathbf{H}_k = {}^{k+1}\mathbf{R}_k + \frac{{}^{k+1}\mathbf{t}_k \mathbf{n}^T}{d_k}. \tag{2.24}$$

${}^{k+1}\mathbf{H}_k$ is a 3×3 matrix, called the homography matrix. Besides, with $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ being the respective projection of ${}^c\mathbf{X}_k$ and ${}^c\mathbf{X}_{k+1}$ into $\mathbf{I}_k$ and $\mathbf{I}_{k+1}$, we have ${}^cZ_k \, \mathbf{x}_k = {}^c\mathbf{X}_k$ and ${}^cZ_{k+1} \, \mathbf{x}_{k+1} = {}^c\mathbf{X}_{k+1}$ and with (2.24) we obtain:

$$\mathbf{x}_{k+1}^T = \frac{{}^cZ_k}{{}^cZ_{k+1}} \, {}^{k+1}\mathbf{H}_k \mathbf{x}_k^T \tag{2.25}$$

$$\mathbf{x}_{k+1}^T \propto {}^{k+1}\mathbf{H}_k \mathbf{x}_k^T. \tag{2.26}$$

Equation (2.25) can also be expressed in pixel coordinates, using (2.18):

$$\mathbf{K}^{-1}\mathbf{p}_{k+1}^T \propto \frac{{}^cZ_k}{{}^cZ_{k+1}} \, {}^{k+1}\mathbf{H}_k \mathbf{K}^{-1} \mathbf{p}_k^T \tag{2.27}$$

$$\mathbf{p}_{k+1}^T \propto {}^{k+1}\mathbf{G}_k \mathbf{p}_k^T \qquad \text{with} \qquad {}^{k+1}\mathbf{G}_k = \mathbf{K} \, {}^{k+1}\mathbf{G}_k \mathbf{K}^{-1} \tag{2.28}$$

The homography matrix ${}^{k+1}\mathbf{G}_k$ is thus a homogeneous transformation from $\mathbb{P}^2$ in $\mathbb{P}^2$, which is defined up to a scale factor and parametrized by 8 degrees of freedom.

2.3 Introduction to pose estimation

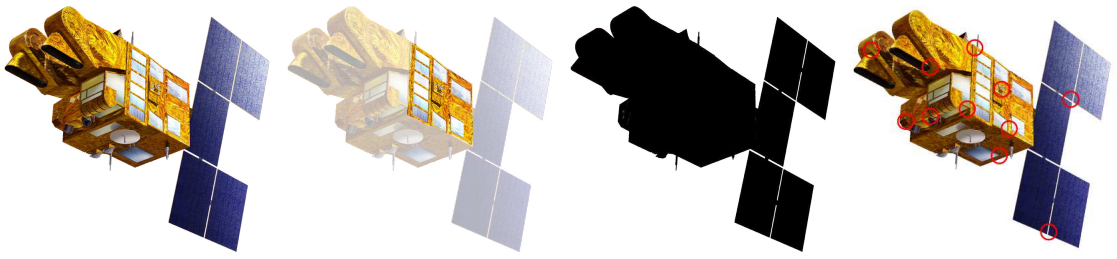
The main objective of this thesis is to compute the 3D rigid transformation between the camera and the target object, given images acquired by the camera. As defined in the previous section, this transformation can be represented by the pose $\mathbf{r}$ or its corresponding homogeneous matrix ${}^{c}\mathbf{M}_{o}$, which can refer to the camera extrinsic parameters.

In order to pertinently address the difficult task of estimating this transformation, using information provided by the projection of the object in images, a first issue to be investigated regards the type of images the problem deals with. Is it a sequence of successive images for which the prior information provided by a frame can be used for the next one or is it a single image? The first case refers to the problem of the frame-by-frame tracking of the object and the second to the problem of the recognition or detection of the object. These problems, which are treated in different manners, are related. For instance in robotic or augmented reality tasks, a visual recognition and detection phase is necessary to initialize a frame-by-frame tracking phase, if no information provided by other sensors can be provided. The next two sections independently present the scopes and the different approaches proposed in the literature to solve these problems. Nevertheless, the same general issues have to be questioned:

- From the available prior knowledge on the 3D object and its appearance, how should the object be represented? Several object representations can be considered in this sense (Figure 2.4): a single or a set of points, a geometrical shape such as a rectangle or elliptical plane, the object 3D model, its silhouette or outline contours, or a higher level single or multi-view appearance model learning visual and geometrical information on the object.

- Which visual information should be processed in the input images? Visual information can be the grayscale values of the object in the image, its colors, its edges extracted from the image, or some textures patches which represent the spatial variations of the intensities of pixels lying in localized regions on the object.

- How this information can be related to the representation of the object to retrieve the searched 3D rigid transformation?



(a) (b) (c) (d)

Figure 2.4 – Examples of geometrical representations of a 3D object, using its 3D model (textured or wireframe) (a), a planar appearance patch or template (b), its silhouette (c), a set of interest points (d).

The next section proposes some insights on how these questions can be answered in the case of pose estimation addressed through a recognition and detection scheme.

2.4 Pose estimation by detection

In robotic tasks involving vision-based pose estimation of the camera with respect to a single known 3D object throughout the task, a key issue lies in detecting the object and determining its pose at the beginning of the task, in order to initialize it. This problem refers to the field of object detection and localization, which faces major challenges when dealing with monocular images. Without any prior hypotheses, how to model and recognize the 3D information from 2D images? The central idea is to learn or extract some information acquired offline on the object, and to match it online with 2D information extracted on the input image. Among the massive literature that addresses this problem, two main categories could be distinguished: global template matching approaches, and local features or descriptor based approaches.

2.4.1 Template matching approaches

The basic idea of template matching, in the case of 3D object localization, is to first acquire a training set of images or views of the object in many different poses. Such views are commonly called templates. Then, at runtime, a similarity measure between the input image and the templates, or the warped templates with respect to some 2D transformations (section 2.2.3), is computed. The template with the highest or lowest score, depending on the type of similarity measure, is selected as the best match, which then enables to retrieve the pose of the camera with respect to the object.

This very early approach has the advantage of being simple but suffers from two major drawbacks: its computational costs and its sensitivity to appearance changes due to illumination conditions, occlusions or viewpoint changes. For decades, many researches have thus focused on efficiently acquiring, organizing and learning the templates and on designing faster and more robust similarity measures.

2.4.1.1 Describing and matching the templates

Let us first review the different ways to describe the templates and match them with the image, through the design of a similarity measure between the template and the image.

Appearance-based features

A classical approach considers the pixel intensities of the template and the input image. These methods can also refer to *appearance matching*. Different similarity measures or alignment functions, coming from information or signal processing theories, with most of them being referenced by [Brown 92] as image matching functions, can be proposed:

- *Sum of Squared Differences* of each pixel intensity:

$$SSD(\mathbf{I}, \mathbf{T}, \mathbf{w}) = \sum_{\mathbf{x} \in \mathcal{T}} (\mathbf{I}(\mathbf{x}) - \mathbf{T}(\mathbf{w}(\mathbf{x}, \boldsymbol{\mu})))^2 \quad (2.29)$$

where $\mathbf{I}$ is the input image, $\mathcal{T}$ a list defining the locations in $\mathbf{T}$. $\mathbf{T}$ is the template and $\mathbf{w}$ is warping function corresponding to an image plane 2D transformation of the template. This basic function lacks robustness to illumination changes, clutter or occlusions. However, linear illumination changes can be modeled and included in the function to improve the robustness [Lai 99].

- Another solution is to use the *Zero-mean Normalized Cross Correlation* or ZNCC defined as:

$$ZNCC(\mathbf{I}, \mathbf{T}, \mathbf{c}) = \frac{\sum_{\mathbf{x} \in \mathcal{T}} \left(\mathbf{I}(\mathbf{x}) - \widehat{\mathbf{I}}\right) \left(\mathbf{T}(\mathbf{w}(\mathbf{x}, \boldsymbol{\mu})) - \widehat{\mathbf{T}}\right)}{\sigma_{\mathbf{I}} \sigma_{\mathbf{T}}}$$

where $\widehat{\mathbf{I}}$ and $\widehat{\mathbf{T}}$ are respectively the average pixel intensities on $\mathbf{I}$ and $\mathbf{T}$, and $\sigma_{\mathbf{I}}$ and $\sigma_{\mathbf{T}}$ are the standard deviations of the two images. However this approach still suffers from sensitivity to non linear illumination changes, occlusions and clutter.

- More elaborate measures have been quite recently experimented such as the *Kernel density* alignment function, which measures a distance between weighted color histograms of both images [Comaniciu 03] and the *Mutual information* (MI), which has been introduced in [Viola 97] as an alignment function and recently revisited by [Dame 12]. They are intended to be much more robust to occlusions and illumination changes, making them suitable and promising for tracking and visual servoing applications. Unfortunately, their computational costs are prohibitive for template matching based recognition and localization tasks.

Edges features

Other classical methods suggest to use image edges as visual primitives or features. Binary edge images can be obtained with a contour detector algorithm such as the one proposed by [Canny 86], for which pixels with the maximum gradient magnitudes in the direction of the gradient are selected as edge points. The principal advantage of edges is that they can be used with many imaging modalities and they offer robustness to changes in sensing conditions such as illumination, noise or blur. *Chamfer Matching* was one of first similarity measure based on edges and was introduced in [Barrow 77] and later developed by [Borgefors 88]. More formally, it is a geometrical measure corresponding to the mean distance between the edge points extracted from the template, and the closest edge points extracted from the image (see Frame 1). Instead of the mean distance, an alternative is to choose the maximum distance, for what is known as the *Hausdorff* measure [Huttenlocher 93, Rucklidge 95] (see Frame 1). The orientation of the edges can also be taken into account for both *Hausdorff* [Olson 97] and *Chamfer* [Shotton 05] measures. As shown by [Olson 97], the number of false positives can be significantly limited and the sensitivity to background clutter reduced.

Frame 1 Classical edge-based similarity measures.

***Chamfer* measure [Barrow 77]**

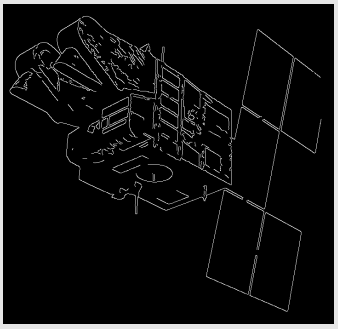
Given the sets of edge points $P_{\mathbf{T}} = \{\mathbf{p}_l^{\mathbf{T}}\}_{l=1}^{N_{\mathbf{T}}}$ and $P_{\mathbf{I}} = \{\mathbf{p}_l^{\mathbf{I}}\}_{l=1}^{N_{\mathbf{I}}}$ respectively extracted from the template $\mathbf{T}$ and the image $\mathbf{I}$, the idea is to look from the points in $P_{\mathbf{T}}$ for the closest points in $P_{\mathbf{I}}$, in terms of the Euclidean distance between pixel coordinates and evaluating the resulting mean distance. The *Chamfer* measure is thus defined by:

$$d_{Chamfer}(\mathbf{I},\mathbf{T}) = \frac{1}{N_{\mathbf{T}}}\sum_{k=1}^{N_{\mathbf{T}}} d_{\mathbf{I}}(\mathbf{p}_k^{\mathbf{T}}) \quad \text{with} \quad d_{\mathbf{I}}(\mathbf{p}_k^{\mathbf{T}}) = \min_{l\in[0\ldots N_{\mathbf{I}}]} \|\mathbf{p}_k^{\mathbf{T}} - \mathbf{p}_l^{\mathbf{I}}\|_2 \quad (2.30)$$

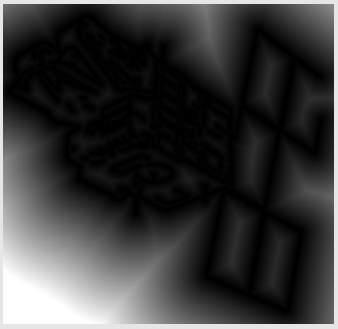
As proposed by [Gavrila 99], it can be efficiently computed by evaluating the distance transform (DT) of the edge map of $\mathbf{I}$.
As depicted on the images below, each pixel in DT is the distance to the closest pixel in the edge map:

$$DT_{\mathbf{I}}(\mathbf{p}) = \min_{l\in[0\ldots N_{\mathbf{I}}]} \|\mathbf{p} - \mathbf{p}_l^{\mathbf{I}}\|_2 \quad (2.31)$$

Input image.



Binary edge map.



Distance transform.

***Hausdorff* measure [Huttenlocher 93, Rucklidge 95]**

$$d_{Hausdorff}(\mathbf{I},\mathbf{T}) = \max_{k\in[0\ldots N_{\mathbf{T}}]} d_{\mathbf{I}}(\mathbf{c}_k^{\mathbf{T}}) \quad (2.32)$$

One drawback of the *Chamfer* and *Hausdorff* measures is their sensitivity to occlusions, since missing edges in the image can make $d_{Chamfer}$ greatly increase. A solution to that shortcoming has been proposed by [Huttenlocher 93, Rucklidge 95] with the partial *Hausdorff* measure, which only computes the maximum of the m^{th} largest distances between the template and the images edge points:

$$d^m_{Hausdorff}(\mathbf{I},\mathbf{T}) = m^{th}_{k\in[0\ldots N_{\mathbf{T}}]} d_{\mathbf{I}}(\mathbf{c}_k^{\mathbf{T}}) \quad (2.33)$$

Shape and silhouette features

Another option is to rely on the shape or silhouette of the object in both the input image and the template. Shape-based methods require a preliminary segmentation step to precisely extract the shape of the object in the template and in the image. An advantage is that the resulting descriptor can be computed relatively to a global position, orientation and scale of the shape in the image, or relatively to finer properties such as stretching or bending. It thus enables some invariance with respect to the corresponding geometrical transformations. For instance shapes can be represented and matched using the medial axis [Zhu 96] and this approach has been extended and enriched in [Kimia 95, Sebastian 01b] with the concept of *shock graphs*, which model the deformations of the shape. In [Sebastian 01a], *curve matching* has been proposed to align two shapes by minimizing a function based on stretching and bending energies between their corresponding curves. Earlier curve-based approaches suggest the use of shape descriptors such as the Fourier descriptor [Persoon 77].

Matching two curves then consists in minimizing the distance between these descriptors. These methods however require the points of the shape to be ordered. Besides they have shown to be sensitive to sampling and to articulations and deformations. Also, [Zhu 96, Kimia 95, Sebastian 01a] based the shapes on the silhouette boundaries of the object, which are invariant to different modalities. However, the case of 3D objects can make these approaches not discriminant enough with respect to viewpoint changes, leading to risks of false positives. In contrast, the *Shape Context* descriptor [Belongie 02] efficiently describes the shape as a set of unordered points, belonging either to the shape outline or to some internal edges (see Frame 2).

However, shape-based methods are still sensitive to occlusion and segmentation errors.

Gradient features

Other researches have suggested to use the image and template gradients as primitives. This dense approach is intended to be more accurate and robust than the previous presented ones. Gradient orientations are indeed invariant to illuminations changes. Besides, densely relying on the template and on the image prevents from problem resulting from segmentation errors, in the edge detection process for intense. For this purpose [Steger 02, Ulrich 09] use the normalized dot product between the template and image gradient vectors. As for intensity-based approaches, these methods can be computationally prohibitive. For a better invariance to some small translations or deformations and a better computational efficiency, a solution is to split both the template and the image into a regular grid and to quantize the gradient orientations into histograms for each grid regions, in the manner of the *Histograms of Gradient* (HoG) feature [Dalal 05]. For real-time performances, [Hinterstoisser 10] based the measure on local dominant gradient orientations, referring to the *Dominant Orientation Template* (DOT) feature. The idea is to split both the template and the image into a regular grid and to only keep, within the regions of the grid, the dominant orientations of the gradients in terms of gradient magnitudes. When the dominant gradient orientation of a region in the image matches one of the dominant orientations in the corresponding region of the translated template to a particular location, the similarity measure increments itself, the idea being to maximize this measure to find the matching location. This measure is made more robust to small translations and de-

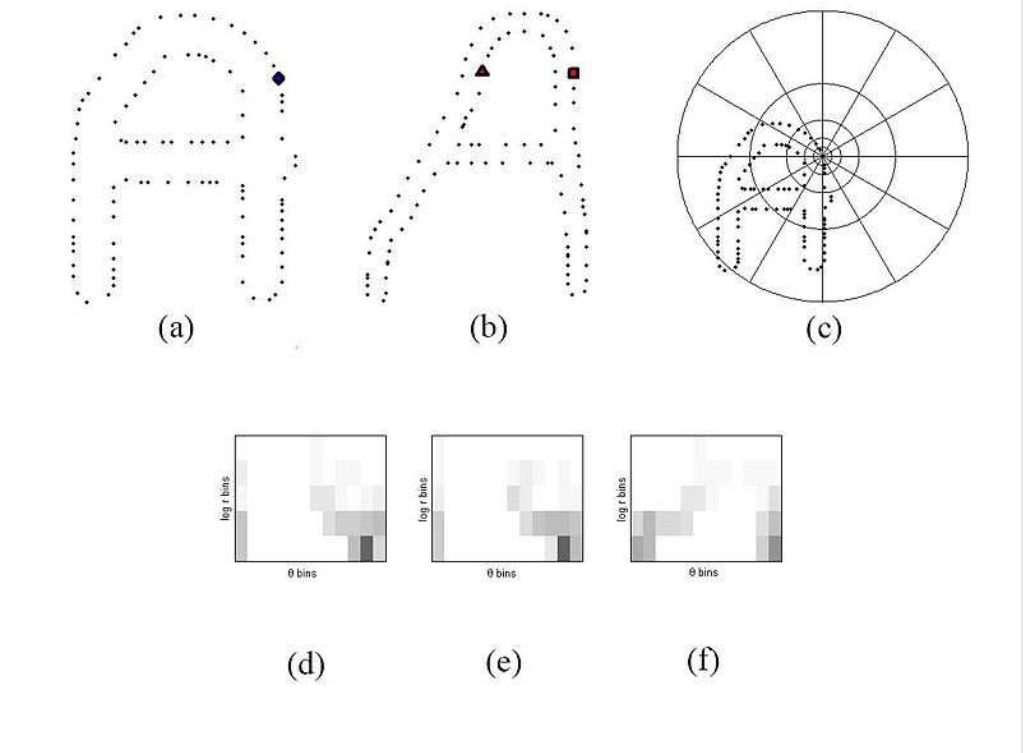
Frame 2 Shape Context descriptor and matching [Belongie 02].

The *Shape Context* consists in computing for each point $\mathbf{p}_i$, belonging the edges of the shape, the vector which connects $\mathbf{p}_i$ to the other points $\mathbf{p}$ of the shape. A histogram is employed h_i for each $\mathbf{p}_i$ to store of the relative coordinates of the remaining points (see Figure).

$$h_i(k) = \sharp\{\mathbf{p} \neq \mathbf{p}_i : \mathbf{p} - \mathbf{p}_i \in bin(k)\} \tag{2.34}$$

Log-polar bins are considered to compute the histogram.

Shape Context



(a) and (b) are the sampled edge points of the two shapes. (c) is the diagram of the log-polar bins used to compute the shape context. (d) is the histogram of the *Shape Context* for the point represented by a square, (e) the one for the diamond, and (f) is for the triangle. The square and the diamond are close points, resulting in similar *Shape Contexts*, in contrast to the triangle.

For matching concerns, two shapes S^0 and S^1 can then be compared at each of their respective reference points $\mathbf{p}_i^0$ and $\mathbf{p}_j^1$ by evaluating the distance $C_{i,j}$ between the corresponding histograms, using the χ^2 test statistic:

$$C_{i,j} = C(\mathbf{p}_i^0, \mathbf{p}_j^1) = \frac{1}{2}\sum_{k=1}^{N}\frac{(h_i^0(k) - h_j^1(k))^2}{h_i^0(k) + h_j^1(k)} \tag{2.35}$$

with $h_i^0(k)$ and $h_j^1(k)$ being the K^{th} bins of the respective normalized histograms of $\mathbf{p}_i^0$ and $\mathbf{p}_j^1$.
Given the set of pair-wise costs $C_{i,j}$ the goal is then to minimize the total similarity measure $H(\pi) = \sum_i(\mathbf{p}_i^0, \mathbf{p}_{\pi(i)}^1)$, where π is a permutation. The result is a permutation π_{opt} and an optimal cost $H(\pi_{opt})$, which is set as the similarity measure between shapes S^0 and S^1.

formations by allowing small independent translations of the regions of the template for a given location in the image. In the same manner [Hinterstoisser 10] made the measure proposed by [Steger 02] more robust to small translations and deformations by allowing small independent translations for each considered pixel in the image, and searching for the maximum value for the similarity measure. Moreover, [Hinterstoisser 10] proposed to take advantage of the parallel architecture of modern computers to accelerate computations.

2.4.1.2 Learning the templates for efficient recognition and detection

As stated before, the goal of template matching is to scan the image with the set of templates and to find the matching location by maximizing or minimizing the similarity measure, with respect to 2D transformation in the image. Since an exhaustive search can be computationally costly, different efficient searching and learning strategies have been conceived to face real-time concerns. In a recognition and localization task, especially for 3D objects, the database can be made of many templates in order to cover the whole space of viewpoints, and the whole space of 2D transformations in the image, through a more or less fine discretization of this space.

Aspect graphs, clustering and classifiers for templates

One way to address this task is to connect the 2D templates or views, generated on a viewing sphere in the case of a 3D object, between each other in order to group or cluster them, with respect to the considered features (appearance, edges, gradients...) and similarity measure, in order to reduce the search space. This concept was first proposed by [Koenderink 76] with the notion of *aspect graph*. The idea is to build a graphical structure of the views for which each node, or *aspect*, represents a cluster, a class of some views. An *aspect* is representative of a connected set of views from which the object visually appears similar.

At the beginning *aspect graphs* were built based on specific types of primitives and objects: lines for polyhedral objects [Stewman 88, Gigus 90, Shimshoni 97], curves for curved objects or solids of revolution [Kriegman 90, Eggert 93]. [Gigus 90] and [Eggert 93] for instance respectively represent lines and curves in terms of *Image Structure Graphs* (ISG) made of junctions between arcs which correspond to lines or curves on the views of the projected 3D object. For these approaches, the changes of appearance of the object between viewpoints are modeled by *visual events* [Shimshoni 97].

However, generating an exact aspect *graph based* on these models is too complex in terms of storage and search requirements. [Cyr 01] extends this concept of aspect graphs by partitioning the views in the set into an *aspect graph* using a shape similarity measures between the views, such as the *curve* or *shock graph* based measures presented above. Each *aspect* is then represented by a *prototype* view. During the recognition phase, the same similarity measure is used to recognize within the database of prototype views the object in the input unknown image. As stated before, this method requires an initial segmentation of the object in both the view or template and the image. Besides, this approach only addresses the task of recognition of an object among others and not its localization.

For appearance-based methods which focus on the grayscale intensity values, a common approach to efficiently learn and group the templates was first introduced in the field of face recognition and makes the use of *Principal Component Analysis* (PCA) [Turk 91]. The central idea is to reduce the set of views, acquired on the viewsphere under different illumination conditions, into a subspace, called the eigenspace, generated by eigenvector of the image.

[Holzer 09] applies the distance transform to the edge-based templates, with the prior that contours in the templates must be closed. This method is designed for recognition concerns between different objects and for localization concerns. Each template corresponds here to a specific object. A *Fern* classifier is trained and used to recognize the object and retrieve its 3D pose. The idea of a *Fern* classifier [Ozuysal 07] is to use simple sets of binary tests, usually comparisons between pixel intensities, in order to returns the probability that an image belongs to any one of the classes that have been learned. Each class is here associated to a template of an object and a pose of this object and the probability of retrieving this class given the tests are learned based on random warps of the template. Despite this method can perform very efficiently, it is limited to planar objects, with closed contours.

Detection and pose estimation

With the aim of efficiently searching within the set of templates, a common strategy is to use a coarse-to-fine searching strategy. The idea is to build offline, in an unsupervised manner, a hierarchy structure or tree of the templates by recursively clustering them. This approach has been adopted by [Rucklidge 95, Olson 97, Gavrila 99, Amit 04, Srivastava 05, Ulrich 09]. [Gavrila 99] uses a *k-means* like clustering technique based on similarity measure between templates relying on the *Chamfer* measure. At each level of the obtained hierarchy, each cluster is represented by a prototype template, which has the smallest distance with respect to the other templates in the cluster. Instead, [Olson 97, Amit 04] build a binary tree of templates, based on oriented Chamfer distance or *Hamming* distance, and each node is represented by a template which stores the overlapping edge pixels between the two clustered templates. In [Ulrich 09], where templates are generated using the CAD model of the object, clusters are recursively built by pairwise matching and merging of templates with neighboring object poses. Similarly, in [Reinbacher 10], the hierarchical clustering method is based on *Affinity Propagation* [Frey 07] and allows some overlap between the viewsphere neighborhoods.
[Hinterstoisser 10] uses a *k-means* like clustering method where templates are recursively clustered by computing the Hamming distance between the cluster template and the remaining templates, until a certain number of templates in the cluster is reached. Since templates are represented as a list of 8-bits integer, the cluster template is computed as a bitwise OR operation applied to the templates of the cluster.

For efficiently searching online within the image transformation space, [Olson 97, Gavrila 99] match their hierarchical structure with the image under a particular 2D transformation which can be iteratively refined when traversing through the hierarchical structure, finally giving a fine transformation and the matching template at the bottom level. For this purpose the corresponding transformation space is decomposed into hierarchical

regular grids. This means that each cell of the grid, which corresponds to a particular transformation, is pruned or selected depending on the similarity measure with respect to the corresponding level of the hierarchical structure. If selected, the cell is then subdivided into a finer grid whose cells are processed with respect to the next hierarchical level of the template clusters.

[Ulrich 09] and [Steger 02] use pyramid resolution levels for both the templates and the input image. In [Steger 02], for each template a resolution pyramid is built and the highest the level of the pyramid is, the coarsest the discretizations of scale and rotation transformations are. At recognition, a resolution pyramid is also built for the input image with the same numbers of levels. Then a breadth-first search through the pyramid levels of the templates is performed, by comparing at each pyramid level each rotated and scaled template, according to the discretization steps, with the input image at the same pyramid level. Potential matches are thus tracked through the pyramid until the lowest level. In [Ulrich 09], this pyramidal approach is combined with the hierarchical clustering of the views, so that each level of the hierarchy is assigned a resolution and the same pyramid is applied to the input image. The range of investigated positions, rotations and scales is refined from one level to the lower one.

Recently, more elaborate training techniques have been experimented for template matching methods for 3D object recognition and pose estimation concerns. In [Gu 10] for instance, which also operates at a object category recognition level, templates are acquired for different object category under different viewpoints. For each category, a discrete set of viewpoints are learned given positive templates showing the object and negative background templates. Each template is described by HOG descriptors. Each viewpoint is then represented by a learned template, in terms of HOG-based feature vectors, of the set of templates classified at this viewpoint. This can be done in a supervised way by labeling each template with their corresponding discrete viewpoint and grouping the templates given their label. A linear Support Vector Machine (SVM) [Schlkopf 02] optimization is performed to learn the reference template of each group of templates. In an unsupervised manner, groups of templates are initialized according to their viewpoint using a Normalized Cut-based clustering technique, using a similarity measure between positive templates based on the HOG descriptor, prior to a SVM optimization. At runtime, a multi-resolution window for each reference template of the discrete viewpoints scans the input image and returns the object category, the viewpoint, the position and the scale of the matched object, based on the dot product of HOG based feature vector.

2.4.2 Local features or part based methods

These approaches aim at learning and classifying local or region features from training images or views, and matching them with 2D features in the image. Knowing the 3D relationship between the learned features, using the 3D model or by learning the underlying spatial relationships, some methods use the resulting 2D-3D correspondences to directly compute the pose or use the matches in a voting process over the pose parameters space in order to determine the most likely viewpoint and image transformation and finally the pose. Though these features are made to handle a wide range of viewing conditions, so to say scale, some affine transformations, illumination changes, a main difficulty remains in

the invariance to viewpoint changes.

2.4.2.1 Identifying and describing local features

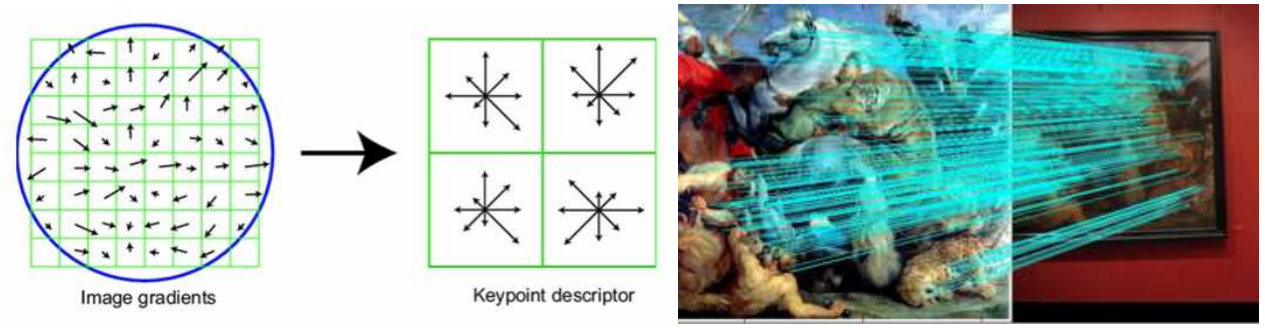
Some local feature detectors

Local features are patches which can be automatically detected or manually selected in the image. When automatically detected, the extraction should be invariant to scale or illumination conditions to enable further matching. A popular detector is the *Harris corner* detector [Harris 88], relying on the eigenvalues of the second-moment matrix. However, Harris corners are not scale-invariant. To circumvent this limitation, [Lindeberg 94] proposed to select scale space local extrema of a *Laplacian-of-Gaussian* (LoG) pyramid or of the scale-normalized determinant of the Hessian matrix. [Mikolajczyk 02] refined this method and created a robust and scale-invariant feature detector with high repeatability, by combining the two approaches proposed by [Lindeberg 94]. [Lowe 99] approximated LoG by *Difference-of-Gaussian* (DoG) to speed up computations. These detectors can thus provide invariance to translation, rotation and scale. In order to cope with invariance to affine transformations, a detector based on the Harris detector along with *affine shape adaptation* can be considered [Baumberg 00, Mikolajczyk 02]. *Maximally Stable Extremal Regions* [Matas 04] can also be an option. [Bay 06] uses for its SURF feature local extrema of the determinant of the Hessian matrix computed on the integral image, speeding up computations. For contours based local features, [Ferrari 08] proposes to detect groups of a specified number of connected contour segments in the image.

Classical feature descriptors

The extracted patches around these detected point are then described using a feature vector, referring to the descriptor, that can deal with different sort of visual information within the patch. Among the numerous descriptors proposed in the literature, a breakthrough has been made by [Lowe 01, Lowe 04] who introduced the *Scale Invariant Feature Transform* (SIFT) which is based on multiple orientation histograms. It has been designed to be invariant to scale, since the scale is determined by interpolating the pyramid of DOG used for detection, and local deformations, with the use of histograms. It is also made invariant to rotation since an orientation of the patch can be retrieved. The patch used for the descriptor is usually divided in 16 regular subregions for which local 8-bins histograms of their gradient orientations are computed (Figure 2.5). It yields a vector of dimension 128, normalized to unit length to provide some tolerance to illumination changes. This descriptor has proven to be very efficient for matching purposes and consequently for 2D and 3D recognition tasks (see section 2.4.2.2). With SURF, [Bay 06] relies on a distribution of Haar-wavelet responses within the patch. For edge-based features [Ferrari 08] registers in the feature vector normalized locations and orientations of the midpoints of the segments with respect to patch centroid, as well segments length.
Some other descriptors have also been designed to describe larger regions, or parts of the object in the image, they are usually not associated to a detector and as a consequence are determined and learned in a supervised way. Among the most significant ones we could distinguish the Histograms of Gradients (HOG) [Dalal 05], which has been previously

presented for global template description purposes. When dealing with edges, Shape Context [Belongie 02] is also commonly used to describe local parts. Also [Shotton 05] uses Oriented Chamfer matching to describe local contour based feature.



(a) (b)

Figure 2.5 – The SIFT descriptor (a), based on the image gradients quantized into orientation histograms, and resulting matching (b), [Lowe 04].

2.4.2.2 Learning the features for efficient recognition and detection

For learning and recognizing in the input image, various methods exist and use local or part features to cope with viewpoint variability.

Among early approaches, [Lowe 01] (Figure 2.6(a)) proposed, in the manner of template matching schemes, to cluster natural training images from similar viewpoints, taken under different illumination conditions, into single model views. Instead of using a global similarity measure for matching the views, this method uses the matching of local SIFT features. Each feature detected in a training image is matched to a feature in the database, which is initialized with the first training image. Using a Hough voting approach, each match for the training images votes for a model view, along with a location, rotation and scale of the matched feature in the model view. The training view is then matched with the model view with most votes. A geometric verification is performed in order to decide whether the training is clustered with model view or not. If clustered, the features in the training image can be added to those referenced for the corresponding model. At runtime, the input image follows the same process so that each of its extracted features are matched to the learned database and casts votes for a model view, along with a location, rotation and scale, giving in the end a most likely viewpoint and a global similarity transform, which can provide a coarse full pose if viewpoints are provided with pose labels with respect the 3D object.

In [Lepetit 06a], interest points are extracted using a Harris corner-like detector on the training images, acquired at different viewpoints and labeled with their 3D positions. These points are then trained using a randomized tree classifier, which, as for *Ferns*, classify the points according to a response to a combination of some binary tests over the local patch of the considered point, viewed under different conditions (affine transformation, illumination). Points extracted on the test image are in the same way matched to trained points and the resulting 2D-3D correspondences are processed in a P3P pose computation algorithm [Fischler 81] (Figure 2.6(b)). Frame 3 presents how pose computation

can be achieved by using 2D-3D point correspondences, with a focus on the PnP problem. This idea of 2D-3D sparse matching has also been studied in [Ozuysal 10, Tola 08] and [Collet 09] which relies on a reconstruction of a 3D metric model using a Structure From Motion technique and whose points are described by local SIFT features. In this field of processing local 2D-3D correspondences, [Lowe 87, Jurie 98, Costa 00, David 03, Strzodka 03] use geometrical features such as straight lines or more complex features based on edges of the object. By matching them, in terms of geometrical distances, with edge features of the 3D model projected with respect to some hypothesized coarse poses, the pose can be refined. As for global template matching techniques which aims at covering the whole pose search space, these approaches can suffer from heavy computational costs since a large amount of hypothesized poses are needed.

Other approaches have addressed object detection, viewpoint classification and thus pose estimation by building a probabilistic model of the object under a discrete set of representative viewpoints, in an supervised or unsupervised manner, by learning parts of the object, or groups of detected local features. [Thomas 06] proposes for instance to use a set of *Implicit Shape Models* (ISM) [Leibe 04] learned at different viewpoints. An ISM corresponds for a given viewpoints to a *codebook* of image patches around interest points detected with Harris corners. These patches, or *codewords*, directly described and matched by their pixel intensities, are clustered across the training images and their appearance and spatial configurations with respect to the object center are learned in a voting process manner. With several ISM under different viewpoints, several corresponding codebooks can be built, thus learning probabilities for location and scale (rotation is not addressed) and also viewpoints of the patches. [Thomas 06] also links the different single view codebooks by tracking features accross training images acquired at different viewpoints. During recognition, the features extracted from the test image are matched to each single-view codebook, casting probabilistic votes for location, scale and viewpoint, and the links between codebooks are used to propagate votes between viewpoints, for a smoother representation. [Ozuysal 09, Glasner 11, Rodrigues 12] are based on similar general ideas. They do not rely on matching detected and learned features but instead propose, as many other 2D or 3D recognition system, a sliding window approach to find potential location and scale of the learned parts of the object at recognition. In that sense, [Ozuysal 09] proceeds in two steps by first training an estimator for the dimensions of the bounding box surrounding the object, and then an estimator of the viewpoint given that bounding box. The related classifier for the viewpoint is trained using a Ferns classifier on spatial pyramids of histograms of clusters of SIFT feature descriptors computed for each pixel within a given bounding box. In [Rodrigues 12], which works at a specific object level, votes here are accumulated at training over the 6D pose parameters, with local patches extracted on image gradients to handle textureless objects, and trained with a Ferns classifier.

[Glasner 11] takes advantage of Structure from Motion techniques (SfM) to reconstruct a 3D point cloud of the object, enabling image patches manually set to be assigned a 3D location, along with a descriptor (HOG computed over a pyramid of spatial bins) for its appearance, a location and a scale. The voting procedure for training and recognition can then be directly handled over the 6D pose parameters, matching between patches appearance being ensured through an SVM classifier. Approaches pro-

posed in [Sun 09, Su 09, Savarese 07] lie in the same category by building 2D multi-part representations and establishing correspondences among the parts, providing a multiview representation. For instance, [Savarese 07] extracts local features out of training images through the Saliency detector, described with SIFT. Assuming the object presents distinctive planar regions, the features are then clustered into planar parts consistently with their appearance and geometry across views from different view points. The object is then represented with as set *canonical parts* whose geometrical relationships (homographies) are determined. At runtime, local features are extracted, clustered given their SIFT descriptors. Obtained parts in the image are matched with *canonical* parts using a search through the test image with a sliding window over position, scale and orientation parameters. Finally the optimal configuration among matched candidate parts enable to retrieve the object class and the pose.

Another line of study [Liebelt 08, Stark 10, Zia 11] proposes to address the problem of multi-viewpoint object representation for object classification and pose estimation by relying on the 3D CAD model of the object, or on a set of 3D CAD models of instances of an object class. [Stark 10] processes edge information by learning labeled parts of the rendered models for each discrete viewpoint, using the Shape Context descriptor (Figure 2.6(c)). As in [Thomas 06, Glasner 11, Sun 09, Su 09] a generative probabilistic spatial model is built and inferred at runtime to retrieve the object class using a sliding window process on the image to match the parts, and to retrieve the viewpoint by matching the best viewpoint dependent spatial model. [Zia 11] extended and refined this method to finer viewpoint classification by, in a similar manner to [Glasner 11], directly incorporating viewpoint parameters in the probabilistic spatial model. Instead of a Hough voting procedure, the inference of the probabilistic model is managed through a particle filter, initialized with the results given by [Stark 10], to retrieve location, scale and viewpoint. [Liebelt 08] instead rely on photorealistic rendered views of the 3D model to build a *codebook* of extracted SURF local features. For each discrete viewpoint features are extracted over different viewing conditions (slight viewpoint variations, illumination) and clustered using *k-means*, while storing the corresponding discrete poses of the features as well as their 3D positions on the object, giving once again a probabilistic spatial model which is inferred in a Hough voting style. During detection local features are extracted on the test image and matched to the codebook, each match casting votes for the stored poses. And the stored 3D positions of the matched codebook entry enables to obtain 2D-3D correspondences which are used to refine the pose using a PnP pose computation solution. All these approach have often been designed for object categorization and detection concerns, proposing coarse viewpoint and pose estimation.

This section presented approaches which aims at computing the 3D pose of the object in a single image, without any coarse a priori, through a global search within a learned training database. This process could be used to initialize pose estimation by frame-by-frame tracking along a video sequence.

2.5 Pose estimation and frame-by-frame 3D tracking

Through frame-by-frame tracking, the idea is to achieve a local search around the pose provided by a frame to compute the pose for the next frame. This problem can actually

Frame 3 Direct pose computation with 2D-3D point correspondences.

Some classical methods allows to compute ${}^c\mathbf{M}_o$ given some known 2D-3D point correspondences.Let $\{\mathcal{X}_i\}_{i=1}^N$ be N 3D points, $\{{}^o\overline{\mathbf{X}}_i\}_{i=1}^N$ their coordinates in $\mathcal{F}_o = (O, \mathbf{x}_o, \mathbf{y}_o, \mathbf{z}_o)$ and $\{\overline{\mathbf{p}}_i\}_{i=1}^N$ their corresponding points in pixel homogeneous coordinates in the image plane. The goal is then, based on equation (2.19) to estimate the projection matrix $\mathbf{P}$ so that:

$$\overline{\mathbf{p}}_i = \mathbf{P}\ {}^o\overline{\mathbf{X}}_i \quad \forall i \qquad \text{with} \qquad \mathbf{P} = \mathbf{K}\Pi\ {}^c\mathbf{M}_o \tag{2.36}$$

The Direct Linear Transform (DLT):

This method [Faugeras 93, Hartley 01] consists in estimating the whole matrix $\mathbf{P}$ by solving a system of linear equations. Indeed 2.36 implies two linearly independent equations and by using a sufficient number a correspondences, the resulting linear system can be solved thanks to Single Value Decomposition (SVD) and coefficients of $\mathbf{P}$ can be estimated. From $\mathbf{P}$ both intrinsic and extrinsic matrices $\mathbf{K}$ and ${}^c\mathbf{M}_o$ can be extracted, but it depends on the geometry of the object and the number of correspondences, usually between 15 and 20 are required.

The Persepective-n-Point (PnP) Problem:

If the camera has been calibrated, *i.e.* if $\mathbf{K}$ has been separately estimated through a calibration method, determining the pose ${}^c\mathbf{M}_o$ requires less point correspondences and can be more stable and reliable for tracking applications than the DLT. A vast literature addresses this problem, known as the P-n-P problem, n referring to the number of point correspondences, and solutions can be classified as non-iterative [Fischler 81, Dhome 89, Gao 03] or iterative [Lowe 91, Dementhon 95, Lu 00]. For instance for the widely studied P3P problem, iterative methods usually consist in estimating the distances $\{x_i = \|O\ \mathcal{X}_i\|\}_{i=1}^3$ and then the points $\{{}^c\mathbf{X}_i\}_{i=1}^3$, using constraints imposed by triangles $O\ {}^c\mathbf{X}_i\ {}^c\mathbf{X}_j$. ${}^c\mathbf{M}_o$ is then retrieved by matching the points $\{{}^o\mathbf{X}_i\}_{i=1}^3$ with the points $\{{}^c\mathbf{X}_i\}_{i=1}^3$. The process generally involves solving an eight-degree polynomial, leading to four solutions in general. The addition of a fourth point enables to remove the ambiguity and obtain a unique solution, for sure when points are coplanar.

[Fischler 81] reduces the P4P problem to P3P one by taking subsets of three points from the four points and checking consistency, introducing the RANSAC procedure (see Frame 7). More recently [Lepetit 09] proposed a more accurate and much less complex method for $n \geq 4$. Iterative methods involve minimizing an appropriate criterion, representing reprojection errors like for the POSIT algorithm [Dementhon 95] or an error expressed in 3D space [Lu 00]. Some approaches such as [Lowe 91] require an approximate solution to initialize the non-linear minimization process. These last iterative methods tend to be more accurate than non iterative ones but are costlier and may not be applied when the points are coplanar.

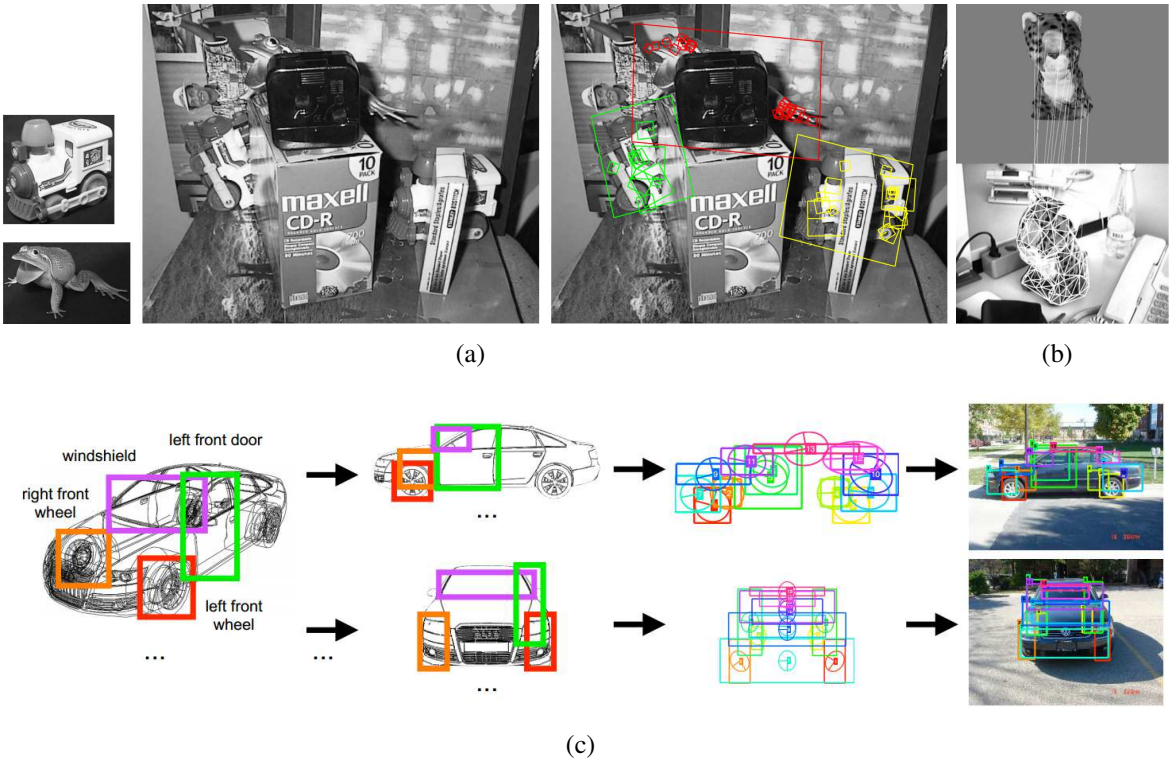


Figure 2.6 – 3D object recognition and pose estimation based on matching local SIFT feature points [Lowe 04](a), using randomized trees [Lepetit 06a](b) and part descriptors on the rendered wire-frame CAD model, using Shape Context [Stark 10](c).

be handled by either directly computing the pose or by computing the displacement from an image to the next one. Among the numerous researches which have addressed the problem of frame-by-frame tracking, the general idea will be to find a way to describe the appearance of the object consistently over time in the image. A goal will be to find visual features to track the object from one image to the next. By determining correspondences between the 2D visual features and the 3D representation of the object, the 3D rigid transformation between the camera and the object can be retrieved. However, some constraints could alter the visual features and tracking abilities across the image sequence. These constraints can be illumination changes, background clutter, occlusion of some parts of the object in the image, image noise, image blur etc.

In order to properly tackle the problem, several issues specific to pose estimation by frame-by-frame tracking should be considered:

- In contrast to detection methods (see section 2.4) for which a global search is necessary, the pose estimation problem can be made local with respect to the pose parameters. From one frame to the next one, the camera motion with respect to the scene is indeed assumed to be small. This way, different suited local estimation frameworks can be carried out (section 2.5.1).

- Given our prior knowledge on the object and its representation, attention has to be paid on the type of visual features which can be tracked between two successive frames, and on the way the correspondences between these features and the repre-

sentation of the object can be determined (section 2.5.2).

- The robustness of the tracking of the features regarding the constraints altering visual consistency of the features from an image to the next (section 2.5.3).

2.5.1 Pose estimation process

The problem of estimating the 3D transformation between the camera and the target from one image to the next one, can be formulated in two different frameworks: a deterministic one and a probabilistic one.

2.5.1.1 A deterministic approach

It consists in, from a known state for frame $\mathbf{I}_k$, to determine the considered geometrical transformation for frame $\mathbf{I}_{k+1}$, through a realignment process between visual features describing the object for $\mathbf{I}_k$ and corresponding visual features extracted in frame $\mathbf{I}_{k+1}$. This is addressed by minimizing an error function using an optimization scheme. Since the problem is non-linear, a non-linear optimization such as like Gauss-Newton or Levenberg-Marquardt, which are presented on Frame 4, is adopted.

Generally speaking, the idea is to estimate a rigid transformation $\boldsymbol{\mu}$ by minimizing, with respect to $\boldsymbol{\mu}$, the least-square non-linear error function $\Delta(\boldsymbol{\mu})$ consisting of errors $e_i(\boldsymbol{\mu}) = f_i(\boldsymbol{\mu}) - b_i$. b_i is a reference observation of a visual feature extracted from the image and $f_i(\boldsymbol{\mu})$ is the value of a corresponding feature resulting, through a mapping function f_i on the observation space, from a projection with respect to the transformation $\boldsymbol{\mu}$:

$$\Delta(\boldsymbol{\mu}) = \sum_i (e_i(\boldsymbol{\mu}))^2 \tag{2.37}$$

- In the case of 2D tracking, $\boldsymbol{\mu}$ can refer to 2D transformations from $\mathbb{P}^3$ in $\mathbb{P}^3$ which can be 2D translations, similarity, affine, or homography transformations.

- In the case of 3D tracking, which we focus on, $\boldsymbol{\mu}$ can correspond to the 3D pose $\mathbf{r}$ between the object and the camera. It can be obtained directly. This pose can also be obtained indirectly by considering as $\boldsymbol{\mu}$ the camera displacement ${}^{c_{k+1}}\mathbf{M}_{c_k}$ between successive frames. This 3D displacement is represented in the image by a 2D transformation, such as a homography, which transfers, as for 2D tracking, the location of the visual features from one location to another in the image. Assuming the first camera pose is known, the current one is obtained by integrating the estimated displacement: ${}^{c_{k+1}}\mathbf{M}_o = {}^{c_{k+1}}M_{c_k}\ {}^{c_{k+1}}\mathbf{M}_o$.

2.5.1.2 A Bayesian approach

It assimilates the pose estimation problem to a statistical one. In this manner, the goal is to estimate the state $\mathbf{x}_k$ of the system, given a dynamic model f of $\mathbf{x}$ to predict $\mathbf{x}_k$ from $\mathbf{x}_{k-1}$ and given some observations or measures $\mathbf{z}_k$. The state model f approximates the

Frame 4 Non-linear optimization.

Since function $f_i(\boldsymbol{\mu})$ is not linear in the case of visual features, the least square error or objective function Δ can be iteratively minimized through non-linear minimization techniques. From an initial estimate $\boldsymbol{\mu}_k^0$, $\boldsymbol{\mu}_k$ is iteratively updated so that $\boldsymbol{\mu}_k^{j+1} = \boldsymbol{\mu}_k^j + \boldsymbol{\delta}^j$ until convergence. The overall goal is to determine at each iteration $\boldsymbol{\delta}^j$ which minimizes $\sum_i (f_i(\boldsymbol{\mu}_k^{j+1}) - b_i)^2$:

$$\boldsymbol{\delta}^j = \arg\min_{\boldsymbol{\delta}^j} \sum_i (f_i(\boldsymbol{\mu}_k^{j+1}) - b_i)^2 \tag{2.38}$$

$$= \arg\min_{\boldsymbol{\delta}^j} \sum_i (f_i(\boldsymbol{\mu}_k^{j} + \boldsymbol{\delta}^j) - b_i)^2 \tag{2.39}$$

Steepest gradient method

In this case, the function Δ is local approximated at a first order by a plane, and to find the optimum, the increment $\boldsymbol{\delta}^j$ is set as:

$$\boldsymbol{\delta}^j = -\alpha \left.\frac{\partial \Delta}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{\mu}_k^j}$$

$$= -2\alpha \left.\frac{\partial s_i}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{\mu}_k^j} (f_i(\boldsymbol{\mu}_k^{j+1}) - b_i) \tag{2.40}$$

$\mathbf{J} = \left.\frac{\partial s_i}{\partial \boldsymbol{\mu}}\right|_{\boldsymbol{\mu}_k^j}$ is the Jacobian matrix of function f_i computed at $\boldsymbol{\mu}_k^j$. α is a scale factor to avoid large steps that would cause the optimization to fall into local minima.

Gauss-Newton method

With the Gauss-Newton method, the function s_i is locally approximated by its first order derivatives, through a Taylor development: $s_i(\boldsymbol{\mu}_k^j + \boldsymbol{\delta}^j) = s_i(\boldsymbol{\mu}_k^j) + \mathbf{J}\boldsymbol{\delta}^j$, and (2.39) becomes:

$$\boldsymbol{\delta}^j = \arg\min_{\boldsymbol{\delta}^j} \sum_i (f_i(\boldsymbol{\mu}_k^j) + \mathbf{J}\boldsymbol{\delta}^j - b_i)^2 \tag{2.41}$$

$\boldsymbol{\delta}^j$ is then estimated so that $\boldsymbol{\delta}^j = -\mathbf{J}^+(f_i(\boldsymbol{\mu}_k^j) - b_i)$, with $\mathbf{J}^+ = (\mathbf{J}^\top\mathbf{J})^{-1}\mathbf{J}^\top$ the Penrose Moore pseudoinverse of $\mathbf{J}$ and $\boldsymbol{\mu}_k$ is updated until a certain convergence criterion is reached. Let us note that the Gauss-Newton method is a particular case of the Newton-Raphson method which consists in approximating function f_i by a parabolic function and to move towards to minimum of this function. With Gauss-Newton, the Hessian matrix used to solve the Newton-Raphson method is approximated by $\mathbf{J}^\top\mathbf{J}$.

Levenberg-Marquardt method

The Levenberg-Marquardt consists in a tradeoff between the Gauss-Newton method and simple steepest gradient descent method, so that:

$$\boldsymbol{\delta}^j = -(\mathbf{J}^\top\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^\top(f_i(\boldsymbol{\mu}_k^j) - b_i) \tag{2.42}$$

λ can be seen as a dumping factor that allows to tune the value of the increment $\boldsymbol{\delta}^j$ given the value of the error function: if at step j Δ does not decrease enough so that Gauss-Newton does not give a descent direction, the idea is to increase λ to take more of a steepest gradient direction. Otherwise λ can be decreased to approach the smoother Gauss-Newton method.

actual dynamic behavior of the system and a noise $\boldsymbol{n}_k$ represents its uncertainty. Observations are also uncertain, what is modeled by a noise $\boldsymbol{w}_k$. Observations are related to the state $\mathbf{x}_k$ through an observation model represented by a function h. This problem can be formalized by the following state and observation equations:

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \boldsymbol{n}_k) \tag{2.43}$$

$$\mathbf{z}_k = h(\mathbf{x}_k, \boldsymbol{w}_k) \tag{2.44}$$

The state $\mathbf{x}_k$ and observations $\mathbf{z}_k$ are thus random variables and determining an estimate $\widehat{\mathbf{x}}_k$ of $\mathbf{x}_k$ given $\mathbf{z}_{1:k} = (\mathbf{z}_1, .., \mathbf{z}_k)$ is equivalent to the statistical problem of determining the probability density function $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$. In the case of visual tracking, the considered state can refer to the 2D or 3D rigid transformations between the camera and the object and also additional parameters such as velocity, jerk... and observations are performed using visual features in the input images. Bayesian filtering, presented on Frame 5 is a general theory to optimally determine $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$, and can be approximated by the widely known Kalman and particle (see Frame 11) filters.

In computer vision, Kalman filters [BarShalom 93] can be considered for tracking and pose estimation of rigid objects [Gennery 92, Harris 92, Koller 93, Yoon 08]. More recently particle filters, introduced in [Isard 98] for visual tracking, have been explored in the case of 3D tracking [Klein 06, Teulière 10, Choi 12]. For such methods, a set of hypotheses on the camera pose is propagated with respect to a dynamic model. The pose is then estimated by evaluating the likelihood of the hypotheses in the image. In [Teulière 10] the particle set is efficiently guided from edge low-level hypotheses.

Both deterministic and probabilistic methods have their advantages and drawbacks. On one hand, iterative non-linear minimization techniques such as Gauss-Newton or Levenberg-Marquardt have the advantage of being fast (only a few iterations are required) and accurate but need to be initialized properly to avoid local minima for which they are sensitive.

On the other hand, Kalman filtering schemes, which are also fast, are useful to stabilize the estimation and to cope with noisy measurements, since they are based on a motion model to predict the state. However, a simple motion model can be unadapted to some particular dynamic scenes, resulting in some lag errors. Besides, measurement and state noise parameters need to be properly determined and tuned. Finally, the considered visual features and observations can be highly non-linear with respect to the pose parameters, making the computation of the Extended Kalman filter tricky. Particle filters, by representing the state by a set of weighted hypotheses can instead deal with highly non-linear problems and more general distributions of both states and observations. But particle filters may suffer from heavy computational costs.

Some recent studies [Teulière 10, Choi 12] have proposed to combine both deterministic and statistical approaches (iterative Gauss-Newton method and particle filtering), gaining the accuracy of the deterministic procedure and the robustness of the probabilistic one, but with a higher computational burden.

Frame 5 Bayesian filtering

Bayesian filtering proposes a general framework to estimate the probability density function $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$ of a state $\mathbf{x}_k$ given observations $\mathbf{z}_{1:k}$. Assuming observations are independent, the a posteriori density $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$ can be recursively computed using Bayes rule from the a priori density $p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})$:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k \mid \mathbf{x}_k)p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})}{p(\mathbf{z}_k \mid \mathbf{z}_{1:k-1})} \tag{2.45}$$

$p(\mathbf{z}_k \mid \mathbf{z}_{1:k-1}) = \int p(\mathbf{z}_k \mid \mathbf{x}_k)p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})d\mathbf{x}_k$ can be seen as a normalization factor and $p(\mathbf{z}_k \mid \mathbf{x}_k)$ is the likelihood of the observations. Assuming $\mathbf{x}_k$ is a Markovian process, equation (2.45) becomes:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \propto p(\mathbf{z}_k \mid \mathbf{x}_k) \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1})p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})d\mathbf{x}_{k-1} \tag{2.46}$$

This recursive formulation can thus be handled through two steps:

- The **prediction step** uses the previous a posteriori probability density $p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})$ and the dynamic model, which enables to compute the density $p(\mathbf{x}_k \mid \mathbf{x}_{k-1})$, to determine the a priori probability density: $p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1})$.

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k \mid \mathbf{x}_{k-1})p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})d\mathbf{x}_{k-1} \tag{2.47}$$

- The **correction step** then provides the new a posteriori probability density $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$, based on the likelihood function $p(\mathbf{z}_k \mid \mathbf{x}_k)$ through equation (2.46).

Bayesian filtering can be applied for any kind of Markovian process with independent observations, whatever the dynamic and observation model and whatever the kind of probability distribution involved. However, the analytical formulations given by equations (2.46) and (2.47) are often inapplicable in some cases. Various solutions have been designed to express this general framework in some particular cases or to approximate it. With linear models and Gaussian distributions, the Bayesian filter is expressed by the Kalman filter.With non-linear models, the Kalman filter can approximated by the Extended Kalman filter (EKF). Finally, Particle filters (Frame 11 provides a general approximation by representing the state with discrete samples.

2.5.2 Visual features

Whatever the optimization approach, a large set of visual features describing the object and tracked across successive frames has been studied and, could be divided into two categories: local geometrical features and global template-based features.

2.5.2.1 Local geometrical features

Edge features

A famous approach to deal with 3D object tracking and pose estimation is to rely on edge features, based on the knowledge of the CAD 3D model of the target. Edge features offer a good invariance to illumination changes or image noise and are particularly suitable with poorly textured scenes, whether the scene is in industrial, outdoor or indoor environments. The main goal is to estimate the pose $\mathbf{r}$ that realigns the edge points generated from the projection of the 3D model with their corresponding edge points extracted from the image.

A widespread idea to handle this problem is the following. Given a new image $\mathbf{I}_{k+1}$, the 3D model of the scene or the target is projected in the image according to the estimated previous camera pose $\mathbf{r}_k$. From each projected edge point $\mathbf{x}_i(\mathbf{r}_k) = pr(X_i, \mathbf{r}_k)$ of the model, a search around the point is performed to find gradient maxima and a corresponding point $\mathbf{x}'_i$ observed in the image. The pose computation is then generally achieved by minimizing the distance between the projected edges of the 3D model and the corresponding edge features in the image. Classically, the error function $\Delta(\mathbf{r})$ can be written as:

$$\Delta(\mathbf{r}) = \sum_i (d(\mathbf{x}_i(\mathbf{r}), \mathbf{x}'_i))^2 \tag{2.48}$$

Among the numerous approaches, various edge-based geometrical features have been proposed to compute the distance $d(.)$ to minimize. An early approach [Brown 71] chose the point to point Euclidean distance whereas [Harris 92, Drummond 02, Marchand 02, Comport 03, Comport 06b, Wuest 07] instead use the perpendicular distance from $\mathbf{x}'_i$ to $\mathbf{x}_i(\mathbf{r})$, given the knowledge of the normal to the edge underlying $\mathbf{x}_i(\mathbf{r}_k)$, since the position of $\mathbf{x}'_i$, simply based on gradient computation, cannot be completely determined, what is known as the *aperture* problem. Other studies explicitly match geometrical primitives such as straight lines or segments, circles... to corresponding primitives extracted in the image. It is the case for [Lowe 92, Gennery 92, Koller 93, Yoon 08] which groups edges extracted in the image into segments which are matched to the projected lines of the CAD 3D model. The matching can be based on the Mahalanobis distance of line segment attributes, such as the coordinates of the middle points, the orientation and the length of the segment [Koller 93]. Once matches have been found, a global error function, computed according to the Malahobnis distance between the projected segments and the extracted ones is also minimized to retrieve the pose.

In order to minimize the error function and to estimate the pose $\mathbf{r}$, these edge-based solutions propose either Kalman filtering [Gennery 92, Harris 92, Koller 93, Yoon 08], particle filtering [Klein 06], or deterministic non-linear minimization techniques [Lowe 92, Drummond 02, Marchand 02, Comport 06b, Tamadazte 10]. The method presented by [Marchand 02, Comport 03, Comport 06b] proposed to turn the minimization problem into an equivalent visual servoing problem by introducing the Virtual Visual Servoing

framework, presented on Frame 6. Other authors have studied a combination of filtering and deterministic approaches [Teulière 10, Choi 12].

Interest point features

Instead of edges, another class of approaches rely on the detection and tracking or matching of interest points. It relies on matching individual features across images. These features can be automatically detected using feature points detection techniques presented in the previous section [Harris 88]. These features can then be represented by local patches around the detected point. Matching a point $\mathbf{x}$ from a frame to a corresponding point $\mathbf{x}'$ in the next frame can be performed through searching on an region surrounding $\mathbf{x}'$ through a similarity measure like the ZNCC [Zhang 95]. An alternative is to use the KLT algorithm to find the translation parameters from $\mathbf{x}$ to $\mathbf{x}'$. Pose estimation can then be addressed through 3D-2D correspondences between 3D points $\mathbf{X}$ lying on a known 3D model of the target, since the 3D coordinates of $\mathbf{x}$ can be retrieved by back projecting to the 3D model.

2.5.2.2 Template-based and dense texture features

Edges or interests points are local features for which edge or point extraction needs to be achieved in the image to derive geometrical 3D-2D correspondences. Another class of approaches instead suggests to rely on dense appearance features, using pixels intensities or more complex dense representations.

- A line of studies has proposed to describe the object pixels with one or a set texture templates or patches, represented by their grayscale values. This class can also refer to template matching techniques which are based on globally tracking of the region of the considered patch or set of patches. It was initially designed for the estimation of 2D transformations [Hager 98, Benhimane 04, Dame 10], with the initial well known KLT algorithm proposed by [Lucas 81, Tomasi 91, Shi 94]. The advantage of this class of approaches is that it dos not require any extraction of visual features. The basic idea is to estimate the displacement of the considered region from one frame to the next one based on a similarity measure between pixels of the reference template or patch and and the image, such the SSD [Lucas 81, Tomasi 91, Shi 94, Hager 98, Benhimane 04], the ZNCC or more elaborate criteria such as the Mutual Information [Dame 10] or the Sum of Conditional Variance [Richa 11, Delabarre 13] to handle robustness to illumination or occlusions. These methods rely on the assumption that the intensity of the projection of 3D points in the image remains constant. This can be done by minimizing the similarity criteria between the image $\mathbf{I}$ and the template $\mathbf{T}$ with respect to the considered transformation $\boldsymbol{\mu}$, through a 2D transformation $\mathbf{w}$ of the template in the image. In the case of the SSD criteria, it can be formulated, similarly to equation 2.29, as:

$$\Delta(\boldsymbol{\mu}) = \sum_i (\mathbf{I}(\mathbf{x}_i) - \mathbf{T}(\mathbf{w}(\boldsymbol{\mu}, \mathbf{x}_i)))^2 \tag{2.52}$$

  Instead of performing a global search on the discretized space of the considered transformation as for template matching purposes, the minimization is then performed through an local non-linear optimization technique such as Newton-Raphson

Frame 6 Virtual Visual Servoing Framework.

Given a new image, the 3D model of the scene or the target is projected in the image according to the estimated previous camera pose $\mathbf{r}$. Each projected line $l_i(\mathbf{r}) = pr(L_i, \mathbf{r})$ of the model is then sampled leading to a set of 2D points $\{\mathbf{x_i}\}$. Then from each sample point $\mathbf{x}_i$ a 1D search along the normal of the projected edge is performed to find a corresponding point $\mathbf{x}'_i$ in the image (see Figure).
In order to compute the new pose, the distances between points $\mathbf{x}'_i$ and the projected lines l_i are minimized with respect to the following criteria [Comport 06b] :

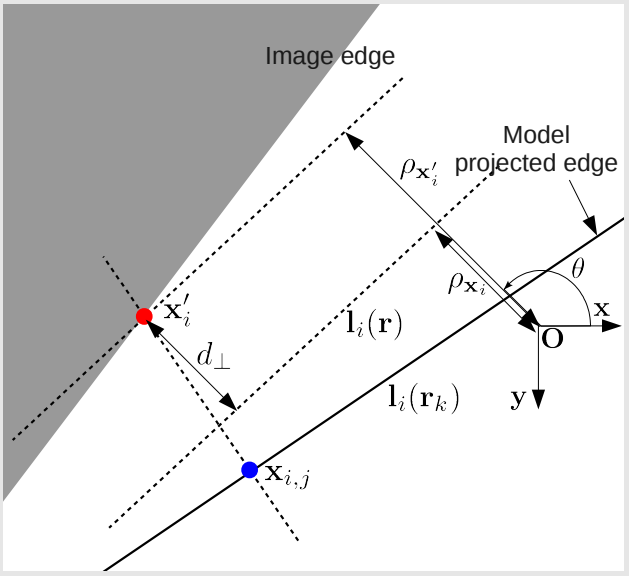
$$\Delta = \sum_i \rho(d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i)) \tag{2.49}$$

where $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i)$ is the distance between a point $\mathbf{x}'_i$ and the corresponding line $l_i(\mathbf{r})$ projected in the image from a pose $\mathbf{r}$. ρ is a robust estimator, which reduces the sensitivity to outliers. This is a non-linear minimization process with respect to the pose parameters $\mathbf{r}$. In this sense, we consider a robust control law which computes the virtual camera velocity skew $\mathbf{v}$ in order to minimize $\mathbf{s}(\mathbf{r}) - \mathbf{s}^*$ and which is given by:

$$\mathbf{v} = -\lambda(\mathbf{D}\mathbf{L}_{d_\perp})^+\mathbf{D}(\mathbf{s}(\mathbf{r}) - \mathbf{s}^*) \tag{2.50}$$

where $\mathbf{L}^+$ is the pseudo inverse of $\mathbf{L}$, the interaction (or Jacobian) matrix of the feature $\mathbf{s}$, which links $\mathbf{v}$ to the velocity of the features in the image. λ is a proportional gain and $\mathbf{D}$ is a weighting matrix associated to the Tukey robust estimator. Finally, the new pose $\mathbf{r}_{k+1}$, represented by its homogeneous matrix ${}^{c_{k+1}}\mathbf{M}_o$, can be computed using the exponential map:

$${}^{c_{k+1}}\mathbf{M}_o = {}^{c_{k+1}}\mathbf{M}_{c_k}\,{}^{c_k}\mathbf{M}_o = e^{-\mathbf{v}}\,{}^{c_k}\mathbf{M}_o \tag{2.51}$$



Moving Edge principle.

or Gauss-Newton. It relies on the computation of the Jacobian matrix $\mathbf{J} = \frac{\partial \mathbf{T}(\mathbf{w}(\boldsymbol{\mu}, \mathbf{x}_i))}{\partial \boldsymbol{\mu}}$, which depends on the gradients of the template $\mathbf{T}$, what can be computationally costly. It can be made faster through some approximations using the conservation of pixel intensity [Hager 98] or off-line learning [Jurie 02]. In its original form, the KLT algorithm was developed to estimate translation parameters, but it has been applied for more complex transformations $\boldsymbol{\mu}$ such as affine [Hager 98] or homography transformations [Jurie 02, Benhimane 04]. As stated before, this last transformation can be integrated to provide the pose. This formulation has also been extended in the case of directly estimating a 3D transformation [LaCascia 00, Jurie 01a, Jurie 01b, Cobzas 05, Panin 08a, Delabarre 13], by minimizing equation (2.52) with respect to 3D camera pose parameters ($\boldsymbol{\mu} = \mathbf{r}$).

But such tracking processes impose small inter-frame displacements and suffers from drift across successive images.

- Some other approaches have introduced optical flow [Horn 87, Heitz 93], which corresponds to the 2D motion of the pixels lying on the object in the image. The principle is to estimate the velocity of these pixels between two successive frames, under the assumption that intensity of the projection of the point remains constant, which can be expressed with the so called optical flow constraint:

$$\frac{\partial \mathbf{I}(\mathbf{x})}{\partial t} dt + \frac{\partial \mathbf{I}(\mathbf{x})}{\partial x} dx + \frac{\partial \mathbf{I}(\mathbf{x})}{\partial y} dy = 0 \tag{2.53}$$

$$\frac{\partial \mathbf{I}(\mathbf{x})}{\partial t} + \frac{\partial \mathbf{I}(\mathbf{x})}{\partial x} \frac{dx}{dt} + \frac{\partial \mathbf{I}(\mathbf{x})}{\partial y} \frac{dy}{dt} = 0 \tag{2.54}$$

Vector $\begin{bmatrix} \frac{dx}{dt} & \frac{dy}{dt} \end{bmatrix}^T = \begin{bmatrix} \dot{x} & \dot{y} \end{bmatrix}^T$ is the optical flow, or apparent velocity, at pixel $\mathbf{x}$, so that the estimated transformation is translation. Though widely used for 2D tracking application [Horn 87, Heitz 93, Mémin 02], this constraint has also been used in the case 3D tracking [Li 93, Basu 96, Pressigout 04].

- Another way to densely describe the appearance of an object is to use its colors. In image processing and computer vision, the RGB color space is widespread to represent color, but other representations such as Luv, $L*a*b$, which are perceptually uniform color spaces, or HSV, which approximates a uniform color space, are also possible.

  Methods presented in [Panin 06, Brox 06, Prisacariu 12] lie in the field of contour-based tracking, relying on the 3D CAD model of the object and restricting to silhouette contours of the projected 3D model in the image. But instead of looking for corresponding edges in the image in terms of gradient maxima as in classical geometrical edge-based frameworks exposed above, these approaches propose to model both side (background and foreground) of the projected contours in terms of color statistics (luminance could also simply be used as stated by [Prisacariu 12]). The principle is then to maximize the separation or segmentation between these statistical foreground and background models, with respect to the full pose parameters. They use posterior membership probabilities [Prisacariu 12] or membership

error function [Panin 06] for foreground and background pixels, and both methods then process their respective energy or error functions with a non-linear optimization technique (gradient descent for [Prisacariu 12], Gauss-Newton for [Panin 06]) to find the optimal pose.

- We can finally mention earlier 3D tracking approaches [Kollnig 97, Marchand 99] which also propose to densely describe edges. They suggest to directly use the gradient in the vicinity of edges and the idea is to estimate the pose for which the gradients on the edges of the projected 3D model best fit the gradients in the image, in terms of SSD on the gradient norms for [Kollnig 97] or in terms of gradient dot product for [Marchand 99]. [Marchand 99] is based on a local exhaustive search on the pose parameter whereas [Kollnig 97] proposes a Gauss-Newton like minimization framework.

All these presented visual features have their advantages and drawbacks. Geometrical edge-based features indeed show robustness against illumination conditions image noise and are particularly suited when the target object is low textured, such as in our spatial context. However, they involve an image extraction process which can lead to outliers and suffer from having similar appearances. It can result in ambiguities between different edges in the image, especially in the case of background clutter. Instead, global template-based visual features, which are based on a richer and more specific description through local patches or regions representing by pixel intensities, or more complex representations, are made to be more discriminative than edges or local interest points. In the case of patch or template tracking, one big issue lies in the choice of the reference template. When taken a priori from a reference image or automatically determined through the detection of interest points, a question is often how the considered patch, template or set of points should be updated or how often feature detection should be performed. Frequent updates would enable to strengthen a spatio-temporal constraint that enables to smooth the estimation process and to account for some illumination, viewpoints changes or transformations not taken into account in the parametrization of the estimated transformation. But it would increase drift in the estimation process due to the accumulation of tracking errors across successive images. Finally, colors offers a rich model for the object appearance which can be very robust to occlusion and background clutter but can suffer from illumination changes in the case spatio-temporal tracking schemes such as [Comaniciu 00, Teulière 09], and can be computationally prohibitive for real-time concerns.

2.5.2.3 Hybrid methods

In order to cope with advantage and drawbacks of different types of cues, some researches have focused on combining them. Different ways of handling the integration can be considered in the literature.

- Some studies propose a sequential integration of edge and texture cues [Bascle 94, Marchand 99], where computed dominant motion [Marchand 99] provides a prediction of the projected edges in the image, improving the initialization the edge-based registration process. [Brox 06] uses optical flow of pixels lying on the projected

object to initialize a maximization process of the separation between object and background along the object contours using color or luminance probabilities, in a similar manner to [Prisacariu 12].

- Other researches simultaneously merge features of different types within probabilistic or filtering techniques such as Kalman filters [Kyrki 05, Haag 99], by integrating interest point matching in an Extended Kalman Filter for [Kyrki 05], and integrating optical flow estimation for [Haag 99]. [Rosten 05] devises a similar hybrid solution within a probabilistic *Expectation Maximization* (EM) framework which aims at optimizing the posterior probabilities of both edge and point features, in terms of distances between the feature projections and determined correspondences in the image. Here feature points are detected through a feature detector derived from corner detection and matched using an SSD similarity measure on pixel intensities on the surrounding patches

- Works of [Masson 03, Vacchetti 04c, Pressigout 06b, Pressigout 08, Panin 08b] rely on a deterministic iterative minimization of a global error function combining the different features. [Vacchetti 04c] proposes to integrate in the classical edge-based error function geometrical distances between feature points detected and matched in two consecutive frames and the projection of their 3D position, retrieved by matching the points to points detected in keyframes with stored 3D positions. The idea is then to simultaneously optimize the reprojection errors in these frames with respect to the 3D position of these points and with respect to the poses for both successive frames, in the manner of key-frame based *Simultaneous Localization and Mapping* (SLAM) techniques. In this case detected feature points are Harris corners, and matching relies on computing normalized cross correlation between patches centered on the detected points. In [Pressigout 06a], in contrast to [Vacchetti 04c, Rosten 05] which consider point to point distances between point features, the texture-based error function is directly based on difference between intensities of some pixels selected on reference planar patches of the object, enabling a simpler parametrization for these point features through homographic transformations. [Pressigout 08] instead relies on optical flow estimation of some regularly spread pixels lying on planar patches on the object, providing point correspondences between successive images and thus point to point distances to be minimized, the transformation for each point from one image to another being also addressed through a homography. [Panin 08b] combine in the objective function to be optimized a classical geometrical information provided by the distances between model and image edges with color information through object/background color separation statistics along the model edges [Hanek 04, Panin 06].

2.5.3 Robust estimation

A stated before, a visual tracking problem can face several constraints such as illumination changes and image noise or blur. This can affect the extraction of some visual features and their matching processes, leading to *outliers* that can impact the quality of the pose estimation process.

Some statistical tools have been used with the objective of reducing the sensitivity of the process to outliers.

2.5.3.1 RANSAC

One of them is the *RANdom SAmple Consensus* (RANSAC) method, introduced in [Fischler 81], and which is a general iterative method to estimate parameters of a model from a set of observed data which consists in "inliers" and "outliers" (see Frame 7). It is particularly suitable for a pose estimation framework. For instance, [Fischler 81], uses it within the P3P algorithm. Triplets of 2D-3D point correspondences are randomly selected are processed to compute the resulting set of pose hypotheses. For each pose, all the 3D points are re-projected in the image and the ones which are sufficiently close to their corresponding 2D points in the image are considered as *inliers* and the pose with the highest number of *inliers* is chosen as the actual pose. This technique has recently been applied by [Bleser 05, Choi 12] in the case of a model-based tracking algorithm. In [Armstrong 95], RANSAC is proposed for 2D-3D line correspondences.

Frame 7 RANdom SAmple Consensus (RANSAC)

RANSAC consists in estimating the parameters μ of a mathematical model based on a set of a set D of N observed data, some of them being *inliers*, data which fit with the model parameters, and others being *outliers*, which do not fit with the parameters. Assuming that n observations are sufficient to compute the model parameters, a number K of subsets of n observations are randomly selected within D. Each subset is used to estimate the model parameters μ_i and to retrieve the corresponding subset $D_i \subset D$ of observations which are consistent which the estimated parameters, based on an error measurement. The subset D_i with the largest cardinal is retained, along with its associated parameters μ_i, from which a least-squares optimization is achieved to refine it, using the points in D_i or the whole set of data D. Several tuning parameters are required during the process, such as the number K of random subsets. [Fischler 81] proposes a formula to determine a coherent value, based on the *inlier* rate. The threshold on the error measurements also has to be set, for instance a few times the standard deviation of the errors.

2.5.3.2 M-estimators

In the case of an iterative minimization of an objective function involving measurements, with respect to some parameters, an alternative is the use of M-estimators, whose goal is to reduce the influence of potential *outliers* in the observations on the estimation process. The idea is by assigning adaptive weights to the observations involved in the objective function (see Frame 8). The concept of M-estimators has been applied to pose estimation [Drummond 02, Vacchetti 04a, Comport 06b, Pressigout 06a, Wuest 07] within non-linear minimization schemes.

Frame 8 M-estimators [Huber 81]

Given an objective function Δ to minimize using an iterative process:

$$\Delta = \sum_i \rho(\mathbf{e}) \tag{2.55}$$

with $\mathbf{e} = \begin{bmatrix} e_0 & e_1 \cdots e_n \end{bmatrix}^T$ an error vector, and $\rho((u))$ a robust function. The general idea is to determine the consistency the errors involved in $\mathbf{e}$, weither they are *inliers* or *outliers*. Instead of binary classification, each error in $\mathbf{e}$ is assigned a weight w_i, with $0 < w_i < 1$, representing the reliability of this error. [Huber 81] proposes an efficient method to compute these weights:

$$w_i = \frac{\psi(\delta_i/\sigma)}{\delta_i/\sigma} \tag{2.56}$$

where $\psi(u) = \frac{\partial \rho(u)}{\partial u}$ is an influence function and $\delta_i = e_i - Med(\mathbf{e})$ is the normalized residue, with Med the median operator. Among the various influence functions in the literature, a popular one in the case visual features is the Tukey influence function [Beaton 74], defined by:

$$\psi(u) = \begin{cases} u(C^2 - u^2)^2 & \text{if } |u| \leq C \\ 0 & \text{otherwise.} \end{cases} \tag{2.57}$$

with C a constant parameter. Parameter σ is a piori unknown but can be estimated online using the *Median Absolute Deviation* (*MAD*) [Huber 81].

2.6 Conclusion

In this chapter, the mathematical background related to monocular computer vision has been introduced. The different involved 3D and 2D geometrical transformations have been set, between the camera and a known scene or a known target object, and within the image. More specifically, since this thesis deals with full localization of the camera with respect to a target object, the issue of pose estimation has been stressed out. The considered pose is the 6 degrees of freedom parameters enabling to fully localize the camera with respect to the target, in terms of position and orientation (or attitude).

Our goal is to design a unified visual localization solution, addressing both problems of tracking the target and initializing the tracking process. Two major lines of studies regarding localization and thus pose estimation have been reviewed. The first one concerns the detection of the target and the estimation of the pose on the initial image. The second one deals with pose estimation through frame-by-frame procedure. Though they are based on similar visual representations of the objects (3D model, interest points, appearance patches...), different concepts and techniques address these issues.

- For **detection and initial pose estimation**, the general idea is to a priori learn some visual information on the object and to match this information with the initial input image, the pose being recovered using the resulting 2D-3D correspondences or relationships.

  Some approaches handle this problem using a global description of the object, by matching a set of views or templates of the object with the input image. These methods refer to template matching. These templates can be learned to reduce the search space and for more efficiency. The matching is based on a similarity measure between the templates and the image, involving different sorts of visual information (appearance, shape, edges...). The pose can be retrieved directly with the stored pose corresponding to the best matching template or indirectly through a voting procedure when templates are learned into a higher-level representation. These methods have the advantage of handling a large range of viewpoints, can be applied to both textured or textureless objects, using natural or synthetic images. They also require few supervision for the learning phase. As a main drawback, they can be limited by their computational efficiency due to the large search space.

  Other approaches address the problem by learning invariant local or regions features, using different sorts of descriptors and by training classifiers or "codebooks" of these descriptors or "visual words", or by building more global multiview appearance models. Then, local or region descriptors extracted from the image are classified within the trained classifiers or appearance models. Finally, using resulting local 2D-3D correspondences, or using a voting process or through geometrical constraints, the pose can be determined. These methods have the advantage that computational aspects are independent of the search space on the pose parameters. Most of these methods are however restricted to textured objects and needs natural training images or photorealistic synthetic views of the object. Besides, the learning phase can require a burdensome amount of supervision.

- With **pose estimation by frame-by-frame tracking**, the aim is, for a frame, to locally search for the pose, or the displacement, that best fits the image content, based on the pose computed for the previous frame. This task can be handled through deterministic non-linear minimization techniques, or through a probabilistic framework, or potentially a combination of both.

  Different visual information or features can be used: edges, gradients, interest points, color, texture patches... with different respective advantages and drawbacks regarding accuracy, robustness and computational efficiency. The attention is particularly paid on solutions based on the 3D model of the target. Hybrid frameworks accounting for these different cues can be considered, benefiting from their complementarity.

Having introduced aspects of both initial visual localization and visual tracking, and reviewed their respective literatures, chapters (3 and 4) describe the solutions we propose to handle these two issues, justifying their respective motivations. In the next chapter, our approach of initial visual localization and pose estimation, using the 3D model of the known target.

Chapter 3

Detection and initial pose estimation

The scope of this chapter deals with our approach of 3D detection and initial pose estimation of a known 3D object in a monocular image sequence, with a focus on the case of a space object, such as a satellite or a debris moving in outter space. This is a key requirement to initialize a robust and accurate frame-by-frame 3D tracking phase, which will be described in chapter 4.

As seen in section 2.4, in the field of monocular 3D object recognition, different classes of approaches can be distinguished among the large literature. We have presented some methods based on global template matching using natural training templates of the object. Some of them consider appearance [Ozuysal 07, Hinterstoisser 10, Gu 10], or shape [Olson 97, Gavrila 99, Holzer 09, Payet 11] to represent the object. Many others are based on learning local or semi-local features described by descriptors such as SIFT [Lowe 04] or SURF [Bay 06], contour descriptors [Ferrari 08], region descriptors such as HOG [Dalal 05], extracted from natural training images of the object. The online recognition phase can then provide pose or viewpoint estimates, through a pose computation step based on 2D-3D point correspondences with the 3D model [Lepetit 06b, Collet 09, Ozuysal 10], using a voting process method [Lowe 04, Thomas 06, Ferrari 08, Ozuysal 09, Glasner 11, Rodrigues 12] or planar matching constraints, relying on a learned multiview appearance and geometry information of the object [Savarese 07, Yan 07, Su 09]. But in our context, these methods, based on real training images, are not suitable since natural images of space objects can hardly be obtained prior to the mission itself. Besides, space objects are often poorly textured or prone to specular effects (for instance due to the insulating film on satellites), making the description of templates or the extraction and description of texture-based feature points complicated.

Using the 3D model of the object

Instead, we propose to rely on another class of approaches which learns the geometry or the shape of the 3D model of the object. As stated before, we deal with known industrial objects (spacecrafts, satellites or parts of them), and accurate geometrical CAD models can be assumed to be provided.

As stated in section 2.4, some template matching methods [Ulrich 09, Reinbacher 10], sparse 2D-3D edge feature matching techniques [Lowe 87, Costa 00, Strzodka 03, David 03, Liebelt 08] or multiview learning frameworks based on part or region descriptors [Liebelt 10, Stark 10, Zia 11], suggest to use and learn the 3D model of the target object and its projection. However, approaches proposed in [Lowe 87, Costa 00, Strzodka 03, David 03], based on matching geometrical primitives such as lines, can be computationally prohibitive, due to the large search space. Furthermore, they face problems when extracting the considered geometrical primitives from edges in the image with degraded conditions such as noise, blur, or background clutter. Region or part descriptor based methods [Liebelt 10, Stark 10, Zia 11] have recently proposed to overcome the issue of computational costs by efficiently learning the 3D model. Though being elegant, these solutions still require a certain amount of supervision during the learning step and are restricted to a coarse set of viewpoints.

Towards template matching

With the aim of designing an unsupervised, and multi-viewpoints method, precise enough to correctly initialize a frame-by-frame tracking process, while keeping computational costs reasonable, we propose to follow the idea of template matching. As reviewed in section 2.4.1.1, some efficient edge or shape based global similarity measures [Olson 97, Steger 02, Belongie 02] have been worked out to cope with occlusion, clutter, noise, specular effects... Our idea is thus to match an exhaustive set, over the 6D pose parameters, of non photorealistic synthetic views of the object. In order to circumvent the problem of the large search space, we propose to rely on efficiently learning, with few supervision, the set of views, as presented in section 2.4.1.2. In this sense, the concept of aspect or view graph [Cyr 04, Toshev 09] or of hierarchical view graph [Ulrich 09, Reinbacher 10], leading to sets of reference views of the object, has aroused our interest. Our pose estimation process can then be performed by matching the input image with these graph structures.

Benefiting from foreground/background segmentation

Besides, our context deals with a single object, moving with respect to the camera located on the chaser. Since a dark uniform (deep space) or cluttered (earth surface) background is assumed, we suggest to take advantage of a foreground/background segmentation technique, as in [Toshev 09]. We propose to spread our object localization process over a sequence of successive input images. Only reasoning on the first frame could indeed result in a too coarse pose estimate, or would require a more exhaustive and costlier searching process over the pose parameters. With our system, the retrieval of the pose is then based on progressively matching and aligning the synthetic views with a short image sequence. At the end of the process the most likely view is determined and selected, along with the stored pose used to render it. Combining this pose with estimated in-plane rotation, translation and scaling parameters to align the view in the image can provide a pose of the object. Indeed, since the dimensions of the object are assumed to be small relatively to its distance from the camera, a weak perspective projection model can be assumed: an isotropic scaling (equivalent to a translation along the optical axis) precedes an orthographic projection.

In [Toshev 09] the sequence of extracted silhouettes from the segmentation phase is

directly used to match and align the silhouettes with the shapes of the model views, using the Shape Context descriptor. This method thus requires a precise segmentation of the object. Besides, the method is based on silhouette contours. Instead, in this work we propose an accurate edge-based similarity measure which is made robust to segmentation errors, by involving both the segmented and the original images. We also suggest to use the image in-plane translation, rotation and scale of the segmented silhouette to coarsely estimate the similarity transformation (see section 2.2.3) of the considered tested view to guide the matching process and compute the pose. This framework would thus result in a faster process, and would be less sensitive to local minima than [Ulrich 09, Toshev 09], which rely on an costlier coarse-to-fine search over these parameters [Ulrich 09], or an exhaustive probabilistic inference over these parameters [Toshev 09].

Overview of the approach

Our method can be outlined by the following steps, and illustrated by Figure 3.1:

- **Learning step** : based on generated synthetic views of the object, it aims at building a hierarchical model view graph leading to some reference views of the model.
- **Pose estimation step along the image sequence** :
  - Foreground/background segmentation of the object. By computing binary moments of the extracted silhouette, the position of its centroid, its orientation, and its area in the image can be retrieved, providing an initial estimate of the image in-plane translation, rotation and scale transformation parameters of the reference views in the image.
  - With the aim of refining these parameters, particle filtering is performed with respect to them, for each reference view, along the input image sequence.
  - We then determine the most likely model view and an associated estimate of the image similarity transform of the view in the image, providing the complete pose, along the input image sequence.
  - Finally the pose is refined by traversing through the hierarchical view graph.

This chapter is organized as follows. Section 3.1 presents the principles of foreground/background segmentation and the solution adopted for this work. The way model views are generated and the way they are learned into a hierarchical view graph is addressed in section 3.2. Our method is then based on a probabilistic alignment framework to determine the most likely reference view, along with particle filtering for image plane similarity transformation parameters, what is detailed in section 3.3. Some experimental results, on both synthetic and real data, are finally provided in section 3.4.

3.1 Segmentation of the moving target object

We deal with a single object moving in an input image sequence, captured using a monocular camera. The aim of this segmentation task is to extract a foreground layer, corresponding to the object silhouette in the image, in the presence of a potentially cluttered

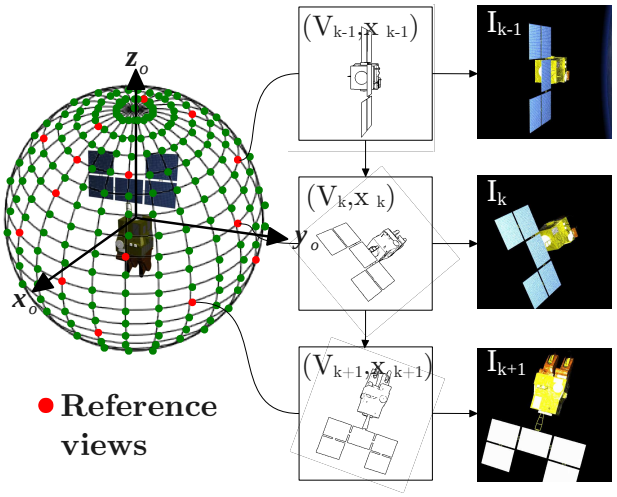
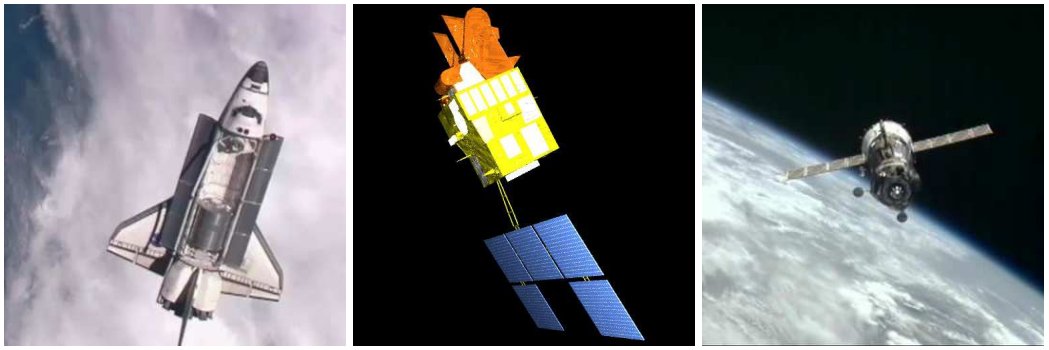


Figure 3.1 – General idea of our pose estimation by detection system. Generated synthetic views of the object are classified into a set of reference views. Reference views are then progressively matched, through V_k, and aligned, through $\mathbf{x}_k$ (the similarity transformation parameters), to the initial images.

and dynamic background. The information provided by the extracted silhouette will further be used in the pose estimation step (section 3.3). With space applications, especially on Low Earth Orbit, three different cases of imaging conditions can be distinguished in our problem, one with the Earth as a dynamic cluttered background, the second with a deep space back background and the third with both earth and deep space as a background, split by the Earth's limb, see Figure 3.2. Two of them ("terrestrial" and "deep space" backgrounds are tackled in this thesis.



(a) (b) (c)

Figure 3.2 – Different types of background in a space context.

3.1.1 A brief review on foreground/background segmentation

Among the vast literature addressing the issue of moving object and foreground/background segmentation, different categories of approaches can be distinguished: inter-frame differ-

ence, background modeling and subtraction, and foreground/background modeling.

3.1.1.1 Inter-frame difference

With the idea of detecting moving pixels, an "old" and simple approach consists in performing difference between successive frames [Jain 79]. The basic concept is then to label pixels with a difference of intensity under a given threshold as background pixels. Choosing the segmentation threshold depends on the illumination conditions and setting a global threshold over the whole frame faces the problem that the contrast can vary on different regions of the moving object. But some researches have tried to overcome these limitations by adapting this threshold to determine wether a pixel has moved or not, as reported in [Konrad 00].

This adaptive threshold can be determined by fitting the difference of intensity between successive images with predefined statistical models corresponding to motion or absence of motion, using hypotheses tests through likelihood ratios. Instead of the difference of intensity, the ratio of intensity can also be used [W. 05]. But performing this thresholding task on each pixel independently can lack of spatial consistency, resulting in noisy segmented maps. This problem can be solved by using spatial information in the determination of the global threshold through Markov Random Fields (MRF) [Aach 95], based on the difference of intensity. However, these approaches are often restricted to static backgrounds.

3.1.1.2 Background modeling and subtraction

Instead of directly using the inter-frame intensity difference, another solution is to build an appearance, spatial or motion model of the background, and to consider pixels which are not consistent with that model as pixels belonging to moving objects of the foreground. Assuming a pure static background, a simple idea of background subtraction is to perform thresholding on intensity difference between the current frame and a reference frame, corresponding to the background. To handle uncertainty on the background appearance, probabilistic models of the background layer have been proposed. The related methods can be predictive or non-predictive. For the non-predictive methods [Kanade 98, Cavallaro 00, Stauffer 99, Zivkovic 04, Elgammal 00, Kim 05], which are the most common, it is based on a probabilistic model of the background using a simple Gaussian distribution of pixel intensities [Kanade 98, Cavallaro 00] or using a mixture-of-Gaussian [Stauffer 99, Zivkovic 04] in order to cope with weakly moving backgrounds, or by estimating a probability density function, such as Gaussian kernels, at each pixel, to handle more complex motion models and dynamic scenes [Elgammal 00]. These approaches however do not take into account some spatial consistency, what is suggested by [Sheikh 05] through the introduction of spatial information in the Gaussian kernels. Another idea has been proposed by [Kim 05], by modeling the background with a *codebook* consisting of some color, intensity, spatial, occurrence information represented by *codewords*. But these methods require a prior knowledge of the background, in the sense that some prior images of the background are needed to model or learn it. Besides, these approaches are still limited to static or weakly dynamic backgrounds.

In order to relax the assumption of a static or weakly dynamic background, requiring the camera to remain almost stationary during the observation, the concept of motion

compensation has been introduced, with the idea of determining a motion model of the background. Detecting pixels belonging to moving objects can indeed be seen as detecting pixels which do not account for the motion induced by the camera. Some works rely on the knowledge of the 3D camera motion, like in [Nelson 91], and the consequent inter-frame 2D flow field in the image is used to determine whether a pixel belongs to this motion or not. However, besides the restricting assumption of knowing the 3D camera motion, these techniques are mostly suited for large inter-frame motions and can result in sparse segmented maps. Instead, another class of methods assumes that the 2D motion in the image of the background, which is due to the 3D camera motion, can be modeled by a 2D parametric motion model [Irani 92, Rowe 96, Odobez 97, Mittal 00, Hayman 03, Ren 03]. A global homography or 2D affine transformation between successive frames is estimated to warp the sequence and to compensate for general motion of the background in the image, extracting the moving objects making of the foreground. These methods are restricted to background which can be approximated as planar or to cameras only subjected to pan or tilt motions. Techniques such as [Irani 92] relies on a thresholding procedure on the warped sequence to determine the foreground layer, leading to noisy or sparse segmentation. [Rowe 96, Odobez 97, Mittal 00, Hayman 03] incorporates motion compensation within appearance statistical background modeling such as Gaussian models [Odobez 97, Rowe 96, Mittal 00] or GMMs [Hayman 03]. [Odobez 97, Hayman 03] also integrate spatial consistency through MRFs.

We can also mention predictive background subtraction methods, for which the idea is to predict the intensity or color of a pixel based on previous observations, through filtering techniques such as Kalman filters [Karmann 90, Koller 94].

Only modeling the background can however be enriched by also modeling the foreground.

3.1.1.3 Statistical foreground/background modeling

This idea refers to the concept of multiple layers segmentation, for which the scene can be decomposed into different layers, each of them having a particular motion model.

More precisely, the idea is to extract and classify these motion patterns, and to model the layers of moving objects and the layer of the background, using probabilistic appearance or spatial models. Based on these models, each pixel can be labeled to one of these layers. When dealing with a single foreground moving object over a potentially dynamic background, as in our context, the goal is to separate two different motion layers and then to build a statistical model for each of these two layers. The labeling task for the different layers can be efficiently achieved by an expectation-maximization algorithm (EM) or using graph cuts [Boykov 01] (see Frame 9). A limitation of these approaches is their lack of computational efficiency, and an initial coarse guess of both layers can be required. Methods in [Criminisi 06, Yin 07] propose real-time solutions, in the case foreground/background layers, but still rely on an offline learning step to get motion priors.

[Xiao 05a, Pundlik 06, Bugeau 07, Sheikh 09] suggest an automatic initial extraction and clustering of the motion layers without learning step. They rely on local feature points which are tracked across frames. Points with similar consistent motions (or trajectories) are grouped together. In [Xiao 05b, Pundlik 06], each extracted point, using the Harris corner detector, is initially assigned a layer. Then, the points are tracked frame-to-frame,

using the KLT algorithm for [Pundlik 06], the method of [Xiao 03] for the approach proposed in [Xiao 05b]. This tracking process provides the apparent motions of the points, through affine transformations. The different apparent motions are used to cluster the initial layers into larger ones: a point is assigned a cluster if its apparent motion matches the overall motion (affine transformation) of the cluster. Once a point is added to a cluster, the affine motion of the cluster is updated. However, only a sparse multi-layer segmentation is provided in [Pundlik 06]. In [Xiao 05b] the clustered layers are used to build statistical models (Gaussian model on the color of the pixels), to achieve a dense complete segmentation. First, as in [Xiao 03], the local points which are clustered into the different layers are expanded, providing larger regions around them. This preliminary step is achieved using binary graph-cuts and its goal is to better describe the models of the layers. Based on the models, graph cuts (Frame 9) are then used to segment the whole set of pixels of the frames. However, these methods [Xiao 05b, Pundlik 06] suppose that the scene can be approximated by a set of planar regions. Let us note that the approaches in [Pundlik 06, Xiao 05b] are suited for the general case of multiple motion layers.

Methods in [Bugeau 07, Sheikh 09] focus on the particular case of foreground / background segmentation. But they generalize this idea of characterizing the motions of the layers (two in that case) to any kind of scenes, with non necessarily planar regions and with potentially dynamic backgrounds. In [Bugeau 07] the foreground can consist of several moving objects, which have consistent motions and colors. The approach is based on the assumption that the motion of background is dominant. Points regularly selected over the frame are tracked using KLT and clustered in to foreground and background layers using a Mean Shift algorithm. Based on a color and spatial probabilistic models of these regions, segmentation is performed using energy minimization via graph cuts. To identify the background, the method in [Sheikh 09] is also based on the assumption that the background is the dominant rigid entity in the image and that it is stationary in a world frame. This last assumption means that the apparent motion in the image sequence depends only the 3D structure of the scene and the motion of the camera (see section 3.1.2.3).

3.1.2 Segmentation in the case of a terrestrial background

Let us remind that based on an initial image sequence, the first goal of our solution is to extract the moving target object from the background. In this section we address this problem in the case of a "terrestrial" background, for which the background consists in the earth surface, as seen on Figure 3.2(a).

3.1.2.1 Motivations for the segmentation approach

As reviewed in the previous section, different foreground/background methods can be considered to handle our problem. Since the apparent motions of both the foreground (the moving object) and the background can potentially be identified, our basic idea is to use a statistical foreground/background modeling technique. In this particular case of a terrestrial background, several assumptions can be made. Indeed, the apparent motion in the image due to the Earth self rotation can be neglected with respect to the apparent motion due to 3D motion of the chaser spacecraft. Local motions such as the local motion of the clouds can also be neglected. We can thus assume that locally the Earth is a rigid

body at rest in space, and that the apparent motion of the background is due to the camera selfmotion. For this reason we can rely on the idea suggested in [Sheikh 09] to identify the background and then to statistically model both background and foreground layers.

Let us first set up the mathematical framework we have adopted to handle the foreground/background segmentation problem, which consists in labeling each pixel of the image to the foreground or to the background layer.

As in many layer segmentation methods, with some of them reviewed in section 3.1.1, we use an energy minimization framework, based on statistical models of the foreground and the background, whose constructions in this work are reported in section 3.1.2.3.

3.1.2.2 Energy minimization formulation

For a an image $\mathbf{I}_k$, we denote by $\alpha = \{\alpha_i\}_{i=1}^N$ the set of the unknown binary labels of the set of pixels $\{\mathbf{p}_i\}_{i=1}^N$ of $\mathbf{I}_k$. $\alpha_i = 0$ when pixel $\mathbf{p}_i$ belongs to the background layer and $\alpha_i = 1$ when it belongs to the foreground layer. Estimating the values $\widehat{\alpha}$ of the labels for an entire image can be formulated as the minimization of an energy-based Markov Random Field objective function $E(\alpha)$, with respect to α:

$$\widehat{\alpha} = arg\min_{\alpha} E(\alpha) \tag{3.1}$$

$$\text{with} \quad E(\alpha) = E_{data}(\alpha) + \gamma E_{smooth}(\alpha) \tag{3.2}$$

with

$$E_{data}(\alpha) = \sum_i D_i(\alpha_i) \tag{3.3}$$

$$E_{smooth}(\alpha) = \sum_{(i,j)\in\mathcal{N}} V_{i,j}(\alpha_i, \alpha_j). \tag{3.4}$$

E_{data} is the "data" energy term, with $D_i(\alpha_i)$ a unitary term which is related to some image "data" (intensity, color, location...) observed in the image at pixel $\mathbf{p}_i$. As its log-likelihood, $D_i(\alpha_i)$ accounts for the observation probability $p(\mathbf{p}_i \mid \alpha_i)$ of pixel $\mathbf{p}_i$ to belong to the foreground or to the background. $p(\mathbf{p}_i \mid \alpha_i)$ is evaluated using the image data at pixel $\mathbf{p}_i$ and the statistical models (in terms of intensity, color, location) previously built for the background and the foreground. More formally, we have:

$$D_i(\alpha_i) = -log(p(\mathbf{p}_i \mid \alpha_i)) \tag{3.5}$$

E_{smooth} is the smoothness energy term, with $V_{i,j}(\alpha_i, \alpha_j)$ a binary term whose goal is to favor smoothness, or spatial coherence within the pixels. $V_{i,j}(\alpha_i, \alpha_j)$ is computed so that it favors coherence on regions with similar pixel intensities or colors. It is determined on sets $\mathcal{N}$ of pairs of pixels which are neighbors. In practice, we consider pixels as neighbors if they are adjacent horizontally/vertically, with 4 or 8-way connectivity. We choose a 4-way connectivity in this work, for computational reasons.

Classically [Rother 04, Criminisi 06], $V_{i,j}(\alpha_i, \alpha_j)$ is computed as:

$$V_{i,j}(\alpha_i, \alpha_j) = [\alpha_i \neq \alpha_j] \frac{e^{-\mu^{-1}\|\mathbf{I}(\mathbf{p}_i)-\mathbf{I}(\mathbf{p}_j)\|^2}}{dist(i,j)}. \tag{3.6}$$

The contrast parameters μ is set as:

$$\mu = 2\ E\left[\|\mathbf{I}(\mathbf{p}_i) - \mathbf{I}(\mathbf{p}_j)\|^2\right] \tag{3.7}$$

where $E[.]$ denotes the expectation over all pairs of pixels $(\mathbf{p}_j, \mathbf{p}_k)$ in an image sample. For $\|\mathbf{I}(\mathbf{p}_i) - \mathbf{I}(\mathbf{p}_j)\|$, we refer to the distance in terms of RGB space color components. $dist(j,k)$ is the distance in terms of pixel image plane coordinates. Let us finally note the fixed scalar γ is a parameter which tunes the balance between the data and smoothness energy terms.

In order to compute the optimal solution of this energy minimization problem and to determine $\widehat{\alpha}$, we employ the *graph cuts* algorithm [Boykov 01] (see Frame 9).

In our context, we propose to compute the data energy term using two different terms. One term is obtained through foreground and background statistical modeling (E^m_{data}, see section 3.1.2.3). The other term is computed by modeling the motion of the background and using homography-based motion compensation (E^c_{data}, see section 3.1.2.4). Formally, E_{data} can be rewritten as:

$$E_{data}(\alpha) = \beta E^m_{data}(\alpha) + (1-\beta)E^c_{data}(\alpha). \tag{3.8}$$

β is a weighting parameter ($0 < \beta < 1$). E^m_{data} and E^c_{data} can be derived as:

$$E^m_{data}(\alpha) = \sum_i U^m_i(\alpha_i) = \sum_i -log(p^m(\mathbf{p}_i \mid \alpha_i)) \tag{3.9}$$

$$E^c_{data}(\alpha) = \sum_i U^c_i(\alpha_i) = \sum_i -log(p^c(\mathbf{p}_i \mid \alpha_i)) \tag{3.10}$$

with $U^m_i(\alpha_i)$ and $U^c_i(\alpha_i)$ the corresponding local energy terms. $p^m(\mathbf{p}_i \mid \alpha_i)$ and $p^c(\mathbf{p}_i \mid \alpha_i)$ are the corresponding observation probabilities.

By using these two terms, the underlying idea is, by relying on some assumptions particular to our context, to combine a foreground/background modeling technique with a background subtraction technique, and to benefit from their complementarity.

3.1.2.3 Feature tracking, clustering and foreground/background modeling

In this paragraph, we present our approach of foreground/background modeling, with the aim of computing E^m_{data}. As in [Bugeau 07, Sheikh 09], we propose to identify and describe both foreground and background layers by processing some feature points that are tracked over a certain number of frames and are clustered as background or foreground points, consistently with their motions or trajectories. The different steps of the method are described hereafter. N_h Harris corner points $\{\mathbf{p}^0_l\}_{l=1}^{N_h}$ are detected on the first frame, with $\mathbf{p}^0_l = \begin{bmatrix} u^0_l & v^0_l \end{bmatrix}^T$, in pixel coordinates (Figure 3.3, left). By tracking these points over

the image sequence with the Kanade-Lucas-Tomasi (KLT) [Shi 94] tracker, we obtain, for a frame $\mathbf{I}_k$, a set of trajectories $\{\mathbf{w}_l^k\}_{l=1}^{N_h}$ over a sliding window of size k_w, $1 < k_w < k$, as represented on Figure 3.3. Each $\mathbf{w}_k$ can be written as:

$$\mathbf{w}_l^k = \begin{bmatrix} \mathbf{p}_l^{k-k_w} & \mathbf{p}_l^{k-k_w+1} & \cdots & \mathbf{p}_l^k \end{bmatrix}^T \tag{3.11}$$

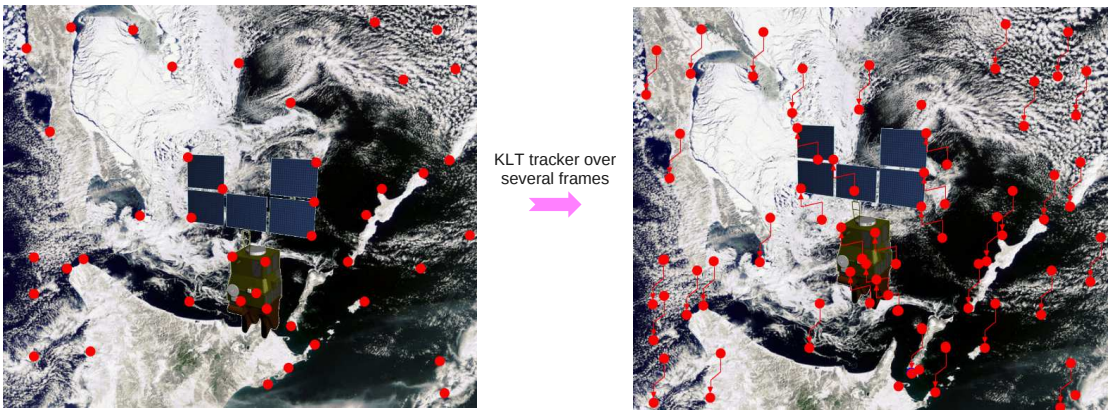


Figure 3.3 – Detection of Harris corners (left, red dots), which are tracked over successive frames, resulting in a set of trajectories (right, red arrows).

The goal is then to cluster these trajectories into background or foreground trajectories. As stated at the beginning of section 3.1.2.1, we can consider, in our application, the background to be stationary in the world frame, so that the apparent motion of the background only results from the 3D motion of the camera. We also assume that the apparent motion of the background is dominant in the image. Based on these assumptions, we follow the approach proposed in [Sheikh 09], which uses the rank-constraint to determine a motion model of the background. This rank-constraint means that the matrix formed by the projected trajectories of stationary points in the world frame is a rank three matrix, so that background trajectories must lie in a subspace spanned by three basis trajectories. More formally, we can first define $\mathbf{W}^k$ as the matrix grouping the set of trajectories:

$$\mathbf{W}^k = \begin{bmatrix} \mathbf{w}_0^k & \mathbf{w}_1^k & \cdots & \mathbf{w}_{N_h}^k \end{bmatrix} = \begin{bmatrix} u_0^{k-k_w} & u_1^{k-k_w} & \cdots & u_{N_h}^{k-k_w} \\ v_0^{k-k_w} & v_1^{k-k_w} & \cdots & v_{N_h}^{k-k_w} \\ \vdots & \vdots & & \vdots \\ u_0^k & u_1^k & \cdots & u_{N_h}^k \\ v_0^k & v_1^k & \cdots & v_{N_h}^k \end{bmatrix} \tag{3.12}$$

Assuming the camera projection is orthographic, the matrix $\mathbf{W}^k$ is a rank 3 matrix. This coarse assumption can be justified in our case since we can suppose the considered proximity or rendezvous operations to be located on the Low Earth Orbit (LEO, where there is the most needs for such operations), so that the surface of the Earth shows few perspective effects. However, as reported in [Sheikh 09], even in sequences with important perspective effects, this assumption does not penalize the consequent segmentation method.

Let us justify the rank constraint asserting that $\mathbf{W}^k$ is a rank 3 matrix under orthographic projection (the reader can refer to [Sheikh 09] for more details). $\mathbf{W}^k$ can indeed

be seen as the projection, on the set of the k_w successive images, of a set of 3D points $\{\mathbf{X}_l\}_{l=1}^{N_h}$, each $\mathbf{X}_l = \begin{bmatrix} X & Y & Z \end{bmatrix}^T$, in a world reference frame, lying on the surface of the background. With an orthographic projection model $\mathbf{W}^k$, the projection is performed using an orthogonal matrix $\mathbf{\Pi}_o$, so that:

$$\mathbf{W}^k = \mathbf{\Pi}_o \begin{bmatrix} X_1 & \cdots & X_{N_h} \\ Y_1 & \cdots & Y_{N_h} \\ Z_1 & \cdots & Z_{N_h} \end{bmatrix} \tag{3.13}$$

$$\text{with } \mathbf{\Pi}_o = \begin{bmatrix} q_1^{k-k_w} & q_2^{k-k_w} & q_3^{k-k_w} & \\ q_4^{k-k_w} & q_5^{k-k_w} & q_6^{k-k_w} & \\ \vdots & \vdots & & \vdots \\ q_1^k & q_2^k & q_3^k & \\ q_4^k & q_5^k & q_6^k & \end{bmatrix} \tag{3.14}$$

Since an orthogonal matrix has full column rank, $\mathbf{\Pi}_o$ is of rank 3 and since the world frame is assumed orthonormal, straightforwardly $\mathbf{W}^k$ is of rank 3. Consequently, each trajectory resulting from the motion of the background lies in a subspace spanned by three basis trajectories:

$$\mathbf{w}_l^k = \sum_{p=1}^{3} a_i \, \overline{\mathbf{w}}_p^k \tag{3.15}$$

with $\overline{\mathbf{w}}_p^k$ the p^{th} basis trajectory. RANSAC is used in order to robustly determine these three basis trajectories from the set of all trajectories. For this purpose, random triplets of trajectories $\mathbf{w}_l^k$, $\mathbf{w}_m^k$ and $\mathbf{w}_n^k$ are selected on the set $\{\mathbf{w}_l^k\}_{l=1}^{N_h}$ to form a projection matrix. This projection matrix is used to compute the projection error of each trajectory $\mathbf{w}_l^k$. Once, based on defined threshold, enough *inliers* are found for a triplet, the process is stopped. Otherwise another triplet is selected and the process is repeated until convergence and the resulting triplet is selected as the triplet of basis trajectories. The projection on this basis is performed for the whole set $\{\mathbf{w}_l^k\}_{l=1}^{N_h}$, finally identifying trajectories which lie within the resulting subspace and which do not.

As a result, this method enables to efficiently cluster trajectories $\{\mathbf{w}_l^k\}_{l=1}^{N_h}$ and the corresponding feature points $\{\mathbf{p}_l^k\}_{l=1}^{N_h}$ into background trajectory points and non-background (i.e. foreground) trajectory points (see Figure 3.4).

These trajectory points $\{\mathbf{p}_l^k\}_{l=1}^{N_h}$, given their membership to the background or foreground layers, are then used to model these layers. However, relying on the whole set can be biased since these points can be concentrated on some particular regions in the image.

Instead, since no a priori are given regarding the shape or the appearance of the moving object or the background, we propose to restrict to trajectory points that are regularly spread in the image plane. In this sense we set a grid G_b for the background and a grid G_f for the foreground:

$$G_b = \{\mathbf{p}_{p,q}^b = (\frac{p.w}{N_b}, \frac{q.w}{N_b}); p, q = 1 \cdots N_b\} \tag{3.16}$$

$$G_f = \{\mathbf{p}_{p,q}^f = (\frac{p.w}{N_f}, \frac{q.w}{N_f}); p, q = 1 \cdots N_f\} \tag{3.17}$$

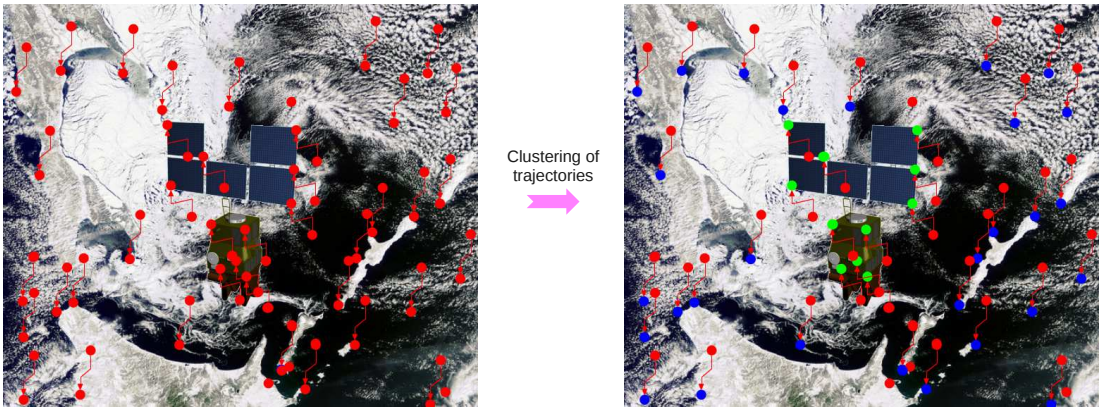


Figure 3.4 – Clustering trajectories into background and foreground trajectories, leading to a set of background points (blue dots) and foreground points (green dots).

We choose the trajectory points that are the closest to the nodes $\mathbf{p}^b_{p,q}$ and $\mathbf{p}^f_{p,q}$ of the grids, and finally we obtain a set of background trajectory points $\{\mathbf{p}^b_i\}_{i=1}^{N_b}$ and a set of foreground trajectory points $\{\mathbf{p}^f_i\}_{i=1}^{N_f}$.

We then use these sets to determine the statistical models of both background and foreground layers. For both layers we suggest, as in [Bugeau 07, Sheikh 09], to use Kernel Density Estimation as probabilistic modeling. But in our approach, we propose to use only a color model for the background. The foreground model is instead based on both color and spatial information. A reason for this choice is that the foreground layer is likely to be concentrated in the image, making spatial information more discriminative than for the background.

More formally, the background model is based on the set of vectors $\{\mathbf{z}^b_i\}_{i=1}^{N_b}$, where $\mathbf{z}^b_i = \begin{bmatrix} R_i & G_i & B_i \end{bmatrix}^T$, with R_i, G_i and B_i the RGB color coordinates of pixel $\mathbf{p}^b_i$. Similarly, the foreground model is based on the set of vectors $\{\mathbf{z}^f_i\}_{i=1}^{N_f}$, where $\mathbf{z}^f_i = \begin{bmatrix} R_i & G_i & B_i & u_i & v_i \end{bmatrix}^T$, with R_i, G_i and B_i the RGB components of $\mathbf{p}^f_i$ and u_i and v_i the pixel coordinates of $\mathbf{p}^b_i$.

Then the probability for a pixel $\mathbf{p}_i$ to belong to the background is then computed using Kernel Density Estimation, by selecting the appropriate data $\mathbf{z}_i$ (RGB or RGB+location) on $\mathbf{p}_i$:

$$p^m(\mathbf{p}_i \mid \alpha = 0) = \frac{1}{N_b \|\mathbf{B}_b\|^{\frac{1}{2}}} \sum_j \phi({\mathbf{B}_b}^{-\frac{1}{2}}(\mathbf{z}_i - \mathbf{z}^b_j)). \tag{3.18}$$

We proceed the same way for the foreground:

$$p^m(\mathbf{p}_i \mid \alpha = 1) = \frac{1}{N_f \|\mathbf{B}_f\|^{\frac{1}{2}}} \sum_j \phi({\mathbf{B}_f}^{-\frac{1}{2}}(\mathbf{z}_i - \mathbf{z}^f_j)) \tag{3.19}$$

where ϕ denotes a kernel function. As in [Sheikh 09], we use the Epanechnikov kernel function. Though it is less representative of the true distribution than a Gaussian kernel, it sill provides decent results and it is much faster to compute. Computational efficiency is here to be considered since kernels are computed for each model vector $\mathbf{z}^b_i$ and $\mathbf{z}^f_i$, for each pixel $\mathbf{p}_i$ in the image. The Epanechnikov kernel function is given by:

$$\phi(u) = \begin{cases} \frac{3}{4}(1-u^2) & \text{if } \mid u \mid \leq 1 \\ 0 & \text{otherwise.} \end{cases} \tag{3.20}$$

$\mathbf{B}_b$ is a 3×3 symmetric positive definite bandwidth matrix, which is manually fixed. For the foreground, $\mathbf{B}_f$ is a 5×5 bandwidth matrix.

As a reminder, the subsequent energy terms are then computed as:

$$U_i^m(\alpha_i) = -log(p^m(\mathbf{p}_i \mid \alpha_i)) \tag{3.21}$$

$$E_{data}^m(\alpha) = \sum_i U_i^m(\alpha_i) \tag{3.22}$$

But using kernel density estimation over all the background and foreground trajectory points can still be computationally expensive, thus both layers can be modeled this way only for the first frame to segment. For the next ones, based on the data provided by this initial segmented frame, the background and the foreground can be instead represented by smoothed color histograms h^B and h^F in the RGB space, which are updated over successive frames, giving:

$$p^m(\mathbf{p}_i \mid \alpha_i = 0)) = h^B(\mathbf{p}_i, \alpha_i) \tag{3.23}$$

$$p^m(\mathbf{p}_i \mid \alpha_i = 1)) = h^F(\mathbf{p}_i, \alpha_i) \tag{3.24}$$

3.1.2.4 Homography-based motion compensation

In the particular case of a terrestrial background and since the potential rendezvous operation would be located on the Low Earth Orbit, with relatively small Field of View for the camera, the Earth surface can actually be approximated as a plane.

Based on this assumption and relying on ideas suggested in previous works presented in 3.1.1.2 such as [Irani 92, Odobez 97, Mittal 00], the idea is to evaluate pixel observation probabilities through motion compensation. It is based on the estimation of the homography transformation induced by the motion of the background in successive frames.

With this motion compensation framework, the idea is to compensate for errors induced by a poor modeling of foreground and the background layers (through the steps presented in section 3.1.2.3), due to misclassification of the trajectory points.

For this purpose we use background trajectory points $\{\mathbf{p}_i^{b,k}\}_{i=1}^{N_b}$ identified at a frame $\mathbf{I}_k$ through the method presented in section 3.1.2.3, and the corresponding points $\{\mathbf{p}_i^{b,k-k_H}\}_{i=1}^{N_b}$ at frame $\mathbf{I}_{k-k_H}$, with $k_H \geq 1$. We use the notation $\mathbf{p}_i^{b,k}$ to stress out the time step k. It was omitted in section 3.1.2.3 for clarity reasons.

Based on the 2D-2D point correspondences between $\{\mathbf{p}_i^{b,k}\}_{i=1}^{N_b}$ and $\{\mathbf{p}_i^{b,k-k_H}\}_{i=1}^{N_b}$, we can compute the homography matrix ${}^k\mathbf{G}_{k-k_H}$ between $\mathbf{I}_k$ and $\mathbf{I}_{k-k_H}$ and the two corresponding planes which approximate the background. This step can be achieved using a RANSAC robust procedure.

RANSAC for homography estimation

RANSAC is classically applied, by randomly selecting sets of 4 2D-2D points correspondences. For a set $\{{}^j\mathbf{p}_i^{b,k}, {}^j\mathbf{p}_i^{b,k-k_H}\}_{i=1}^4$ of 4 correspondences, a homography ${}^k\mathbf{G}_{k-k_H}^j$ can

be computed [Malis 98]. This homography is then applied to the whole set $\{\mathbf{p}_i^{b,k-k_H}\}_{i=1}^{N_b}$. The resulting re-projection errors with respect to the corresponding points $\{\mathbf{p}_i^{b,k}\}_{i=1}^{N_b}$ (in terms of Euclidean geometrical distances between the points) are computed and enable to identify "inliers" and "outliers", based on predefined threshold. If the consensus is reached, the process is stopped. Otherwise another set is selected and the process is repeated until a consensus is found.

Since only background trajectory points are used, few true outliers are actually processed in the RANSAC procedure, for which the consensus is rapidly obtained, giving a consistent estimation of the homography.

Likelihood evaluation

The homography transformation ${}^k\mathbf{G}_{k-k_H}$ is then applied to the whole frame $\mathbf{I}_{k-k_H}$, to compensate for the computed motion between $\mathbf{I}_{k-k_H}$ and $\mathbf{I}_k$, resulting in an error $\mathbf{e}(\mathbf{p})$ defined by:

$$\mathbf{e}(\mathbf{p}) = \mathbf{I}_k(\mathbf{p}) - \mathbf{I}_{k-k_H}({}^k\mathbf{G}_{k-k_H}(\mathbf{p})) \tag{3.25}$$

Thus, the more the color components of $\mathbf{e}(\mathbf{p})$ are close to zero, the more pixel $\mathbf{p}$ is likely to belong to the background. We can model the compensated background apparent motion as Gaussian and likelihoods can then be evaluated by a Gaussian kernel, with a bandwidth σ:

$$p^c(\mathbf{p} \mid \alpha = 0) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\|\mathbf{e}(\mathbf{p}\|)^2}{2\sigma^2}} \tag{3.26}$$

$$p^c(\mathbf{p} \mid \alpha = 1) = 1 - p^c(\mathbf{p} \mid \alpha = 0) \tag{3.27}$$

As a reminder, the subsequent energy terms are then computed as:

$$U_i^c(\alpha_i) = -log(p^c(\mathbf{p}_i \mid \alpha_i)) \tag{3.28}$$

$$E_{data}^c(\alpha) = \sum_i U_i^c(\alpha_i) \tag{3.29}$$

3.1.3 Segmentation in the case of a deep Space background

As noticed on Figure 3.2(b), the background can also be the uniform black deep space. This case can be dealt with quite easily since a simple threshold on the pixel intensities would provide the foreground layer corresponding to the target moving object. However, in order to cope with potential noise or halo due to sun reflection on the target, we propose a more robust solution, ensued from the graph-cut based segmentation method presented above, without the procedure of modeling foreground and background layers with the feature tracking and clustering, which becomes useless, reducing computations significantly. Instead, we use thresholding initially to determine the smoothed histograms h^B and h^F on the RGB color space, modeling the foreground and the background and giving likelihoods:

$$p(\mathbf{p}_i \mid \alpha_i = 0)) = h^B(\mathbf{p}_i, \alpha_i) \tag{3.31}$$

$$p(\mathbf{p}_i \mid \alpha_i = 1)) = h^F(\mathbf{p}_i, \alpha_i) \tag{3.32}$$

Frame 9 Energy minimization via Graph Cuts [Boykov 01].

A segmentation task can be formulated in terms of labeling pixels of the image to a particular layer, by minimizing an energy function depending on some properties of the image. The most famous algorithm to minimize these energy functions has been proposed [Boykov 01], using Graph Cuts, with the *Expansion Move* algorithm. The general idea is to relate the problem of minimizing such energy functions to the problem of maximizing the flow or minimizing the cut of a graph, with respect to the set of labels. The algorithm is iterative and can applied for binary or multi-layer segmentation tasks. We restrict ourself to the binary case here. More formally, let us denote by E the energy function, which is function of a set of binary variables α. α corresponds to the set of labels of the pixels. E can be associated to a graph $\mathcal{G} = (\mathcal{V} \cup \{\mathcal{S}, \mathcal{T}\}, \mathcal{E})$. Each pixel, with its label α_i in α, is associated to a node $s_i \in \mathcal{V}$, with $\mathcal{V}$ the set of intermediate nodes of $\mathcal{G}$, $\mathcal{S}$ and $\mathcal{T}$ being the terminal nodes and $\mathcal{E}$ the arcs of the graph, linking the nodes. $E(\alpha)$ is equal the capacity of a cut in $\mathcal{G}$, a cut being a partition of the nodes of the graph into two disjoint subsets, S for nodes including the terminal node $\mathcal{S}$ and T for nodes including the terminal node $\mathcal{T}$. The capacity of a cut is the sum of the capacities on the arcs lying on the cut. A node s_i belongs to S if $\alpha_i = 0$ and reciprocally it belongs to T if $\alpha_i = 1$. Equivalently, S can be seen as the subset of background pixels and T the set of foreground pixels.
In order to minimize the energy function E and the corresponding capacity with respect to α, the Expansion Move algorithm is run iteratively. Given a current labeling α, the label α_i of a node can be modified only if different from a particular labeling α_p, for instance $\alpha_p = 1$. The α_i is set to α_p if the resulting cut has a bigger capacity than the previous one. For this purpose capacity of arcs should be determined. Arcs $\mathcal{E}$ can actually be divided in two subsets $\mathcal{E}_1$ and $\mathcal{E}_2$. $\mathcal{E}_1$ contains arcs linking $\mathcal{S}$ and $\mathcal{T}$ with the intermediate ones $\mathcal{V}$, and $\mathcal{E}_2$ the arcs linking intermediate nodes between them. Equivalently, E is formulated as follows:

$$E(\alpha) = \sum_i E_{data}(\alpha_i) + \gamma \sum_{(i,j) \in \mathcal{N}_{(i,j)}} E_{smooth}(\alpha_i, \alpha_j) \tag{3.30}$$

with E_{data} a unitary energy term equivalent to the capacities of arcs in $\mathcal{E}_1$. $E_{data}(\alpha_i = 0)$ is equal to capacity of an arc between s_i and $\mathcal{S}$, and reciprocally with $E_{data}(\alpha_i = 0)$ and $\mathcal{T}$. In the image represent a quantity related to the image data depending on the label. It favors the labeling to respect to observations in the image (intensity, color...). E_{smooth} is a binary term equivalent to the capacities of arcs in $\mathcal{E}_2$. In the image, it thus involves neighborhood pixels, It tends to favor the labeling to respect some discontinuities in the image. $\mathcal{N}_{(i,j)}$ denotes each arc linking nodes s_i and s_j or equivalently pixels $\mathbf{p}_i$ and $\mathbf{p}_j$, which mutually neighbors. Usually, 4 or 8 neighborhood nodes are considered. The more the discontinuities (or gradients) between neighboring pixels are important, the smaller the capacities of their arcs are, favoring cuts on these regions.
α can be initialized to 0 (only background pixels) is there can be prior data information corresponding to some background or foreground modeling, which enables to compute E_{data} consistently. Otherwise, some *seed* nodes or pixels, with predetermined labels, have to be set, automatically or manually, and from which *Expansion Move* is performed based on spatial discontinuities between neighboring pixels.

And we set the energy terms:

$$U_i(\alpha_i) = -log(p(\mathbf{p}_i \mid \alpha_i)) \tag{3.33}$$

$$E_{data}(\alpha) = \sum_i U_i(\alpha_i) \tag{3.34}$$

h^B and h^F are then updated given the segmented background layer.

Through this segmentation step, we are able to extract the target moving object (or foreground) from the background, in the case of a "terrestrial" or "deep space" background.

In the next sections, we present the way the extracted silhouettes, corresponding to the target object, from a sequence of initial images, can be used to match and align a set of synthetic views of the object, with the aim of retrieving the complete 6D pose between the camera and the target object.

3.2 Generation and classification of synthetic views

As presented in the introduction, our pose estimation by detection strategy can be affiliated to template matching methods. It is based on the 3D model of the target and relies on matching, aligning synthetic views of the model with a sequence of initial input images. This section focused on the generation of these synthetic views and on how they are classified to efficiently handle the matching process.

3.2.1 Generation of the views on a view sphere

By synthetic views, we mean projections of the 3D CAD model of the object on blank frames. These synthetic views are generated on a view sphere which is centered on the 3D model, and which is parametrized by 3 DoF (azimuth ϕ, elevation ψ and distance d_0 between the camera optical center and the object center). The rendering is managed using a 3D rendering engine by setting virtual cameras at uniformly sampled viewpoints (green dots on Figure 3.5), corresponding to discrete values of azimuth ϕ ($0 < \phi < 2\pi$) and elevation ψ ($0 < \psi < \pi$) angles, at a fixed distance d_0 from the model, looking at the origin of the frame attached to the 3D model (the origin being the barycenter of the vertices making of the model). The fixed distance d_0 is set to a value so that the whole object can be seen in the resulting rendered view.

As previously stated, both the offline classification of the views and the online matching and alignment phases are achieved using a similarity measure: between two synthetic views, for the off-line learning phase, and between a synthetic view and the input image, for the online matching phase.

In the approaches proposed in [Cyr 04, Toshev 09, Reinbacher 10], which deal with a similar idea of classifying a set of synthetic views of the object, the shapes or silhouettes of the object in the synthetic views are processed. More precisely, a shape-based similarity measure is used in [Cyr 04] (through shock-graphs), the Shape Context, computed on the set of silhouette edges, is adopted for [Toshev 09], and Normalized Cross Correlation between silhouette edges is performed in [Reinbacher 10].

Instead, we propose to deal with the whole set of edges of the views (and of the image in the online phase). Both silhouette and internal edges of the rendered views are extracted, through a Laplacian filter computed on the depth maps of the rendered views (see section 4.2.1), in order to describe more accurately the geometry of the object (see Figure 3.6).

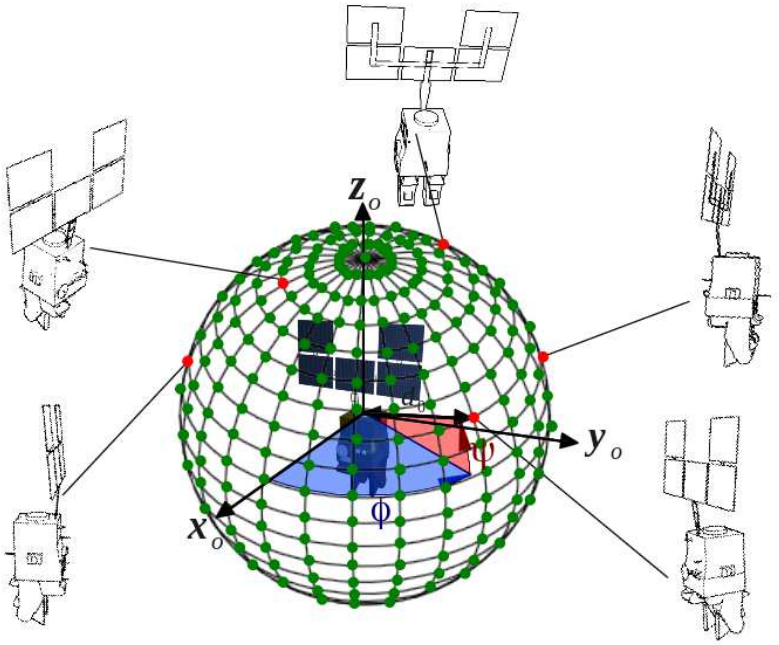


Figure 3.5 – Generation of synthetic views on a view sphere centered on the 3D model at regularly sampled viewpoints, parametrized by their azimuth (ϕ) and elevation (ψ) angles, and by the radius d_0 of the sphere. Edges are extracted by processing the depth buffer of the rendered 3D model.

For each generated view V, we store the pose ${}^{c}\mathbf{M}_{o}^{V}$ used to render the 3D model. Besides, we also store the centroid $\overline{\mathbf{c}}^{V} = \begin{bmatrix}\overline{u}^{V} & \overline{v}^{V}\end{bmatrix}$, orientation α^{V} and area A^{V} of the silhouette of the projected 3D model (see Figure 3.6). These parameters can be evaluated, by using image moments which are computed on the pixels lying within the silhouette of the object:

$$\overline{u}^{V} = \frac{m_{10}}{m_{00}} \quad \overline{v}^{V} = \frac{m_{01}}{m_{00}} \quad A^{V} = m_{00} \tag{3.35}$$

$$\alpha^{V} = \frac{1}{2} arctan\left(\frac{2\left(\frac{m_{11}}{m_{00}} - \overline{u}^{V}\overline{v}^{V}\right)}{\frac{m_{20}}{m_{00}} - \frac{m_{02}}{m_{00}} - \left(\overline{u}^{V\,2} - \overline{v}^{V\,2}\right)}\right) \tag{3.36}$$

where

$$m_{ij} = \sum_{u^V} \sum_{v^V} (u^V)^i (v^V)^j \tag{3.37}$$

$i+j$ being the order of moment m_{ij}, and $\mathbf{p}^{V} = \begin{bmatrix}u^{V} & v^{V}\end{bmatrix}^{T}$ are the pixels lying within the silhouette of the object on view V.

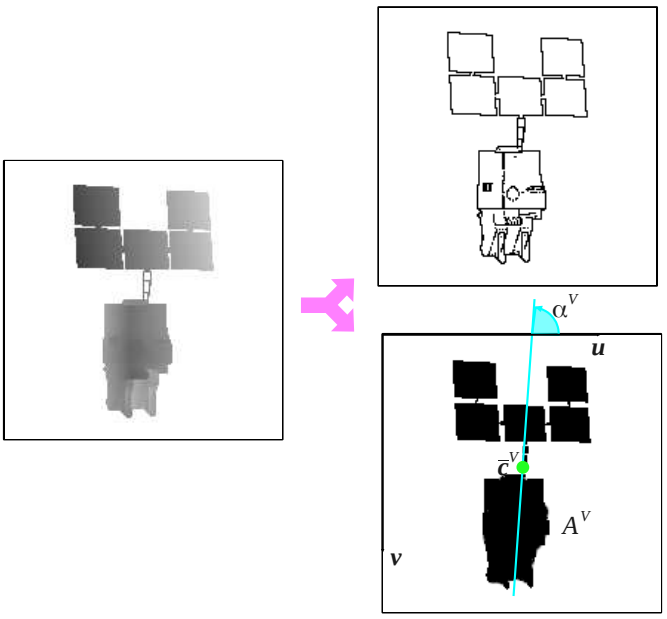


Figure 3.6 – Depth buffer of a rendered view (left) processed into a Laplacian filter to obtain salient edges (top right). The silhouette (bottom right) is also processed to compute moments on the silhouette (black) pixels. Moments enable to retrieve the centroid $\mathbf{c}^0$, the orientation α^V (with respect to the major axis of the silhouette) and the area A^V of the silhouette.

3.2.2 Building a hierarchical view graph and determining reference views

Since the process of matching the whole set of views with the input images can be computationally challenging, we suggest to learn the views by iteratively clustering them into a hierarchical view graph [Olson 97, Gavrila 99, Cyr 04, Ulrich 09, Toshev 09] [Reinbacher 10], as reviewed in section 2.4.1.2 and as it can be described on Figure 3.8. Therefore, we have chosen a similar technique to [Reinbacher 10], using an unsupervised clustering technique based on Affinity Propagation[Frey 07].

To perform clustering, let us first note that we restrict ourselves to a clustering method which selects actual synthetic views as centers of the clusters. Merging or computing means is indeed not well adapted to data such as images or views. With Affinity Propagation, actual data points (in our case views) are selected as centers of the clusters.

As demonstrated in [Frey 07], Affinity Propagation also shows better performances than classical *k-medoids* techniques, especially on large sets of data, with potentially numerous classes or clusters, and particularly in case of clustering image data [Dueck 07]. Besides, with this technique, few parametrization is needed, such as setting the number of clusters (see Frame 10 for an overview of this method).

At the first level of hierarchy, we build clusters within disjoint neighborhoods in the spherical space, in order to cope with memory requirements. This is done by comparing the views with each other in each neighborhood with respect to the considered similarity measure (see section 3.2.3). A slight overlap can actually be considered between the different neighborhoods to consider inter neighborhood variabilities (see Figure 3.7). The

result is a set of clusters, each cluster being represented by a reference view. This set of clusters and reference views establishes the first level of our hierarchical structure. We proceed in the same way with the views of the first level. As this set generally has an acceptable size in our applications, memory problems become non-critical. As a consequence, we do not consider spatial neighborhoods from this level. We can then iteratively build successive hierarchical levels with this method until a reasonable number N_r of reference model views is reached. A set $\{V^j\}_{j=1}^{N_r}$ of reference views is finally obtained.

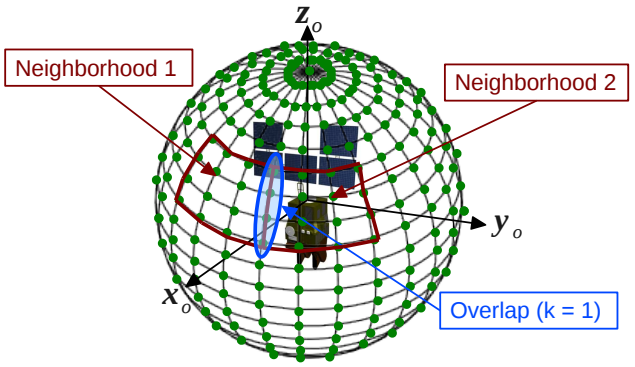


Figure 3.7 – Clustering on the first level within two neighborhoods. A slight overlap (of size $k = 1$) is considered between the neighborhoods.

3.2.3 Similarity measure: oriented *Chamfer Matching*

With the aim of comparing two synthetic views, we propose a first similarity measure D. It is derived from the *Chamfer Matching* distance presented on Frame 1 and on equation (2.30). The idea is to rely on all the extracted edges of the two views, instead of the silhouette contours [Toshev 09, Reinbacher 10]. From the sets of pixel edge points $\{\mathbf{p}_k^i\}_{k=1}^{N_i}$ and $\{\mathbf{p}_k^j\}_{k=1}^{N_j}$ (both silhouette and internal contours) on views V^i and V^j, we compute an oriented Chamfer matching distance by looking for the closest contour from one view to the other:

$$D_{(i,j)} = \frac{1}{2}(d_{(i,j)} + d_{(j,i)}) \tag{3.38}$$

$$\text{with } d_{(i,j)} = \frac{1}{N_i}\sum_{k=1}^{N_i}(d_j(\mathbf{p}_k^i) + \lambda d_j^\theta(\mathbf{p}_k^i)) \tag{3.39}$$

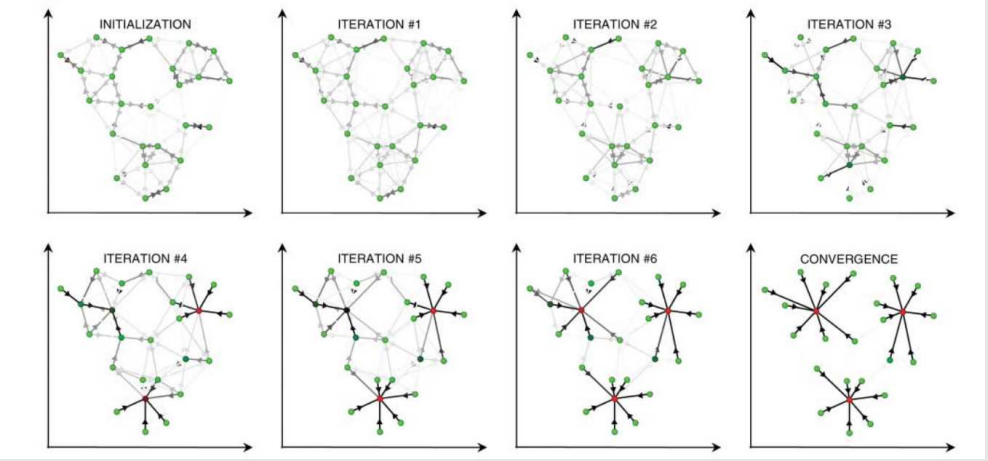
$$\text{where } d_j(\mathbf{p}_k^i) = \frac{\min_{l\in[0\ldots n]}\|\mathbf{p}_k^i - \mathbf{p}_l^j\|_2}{\|\mathbf{p}_k^i - \overline{\mathbf{c}}^i\|_2} \tag{3.40}$$

$$\text{and } d_j^\theta(\mathbf{p}_k^i) = \left|\theta(\mathbf{p}_k^i) - \theta(\mathbf{p}^j_{arg(d_j(\mathbf{c}_k^i))})\right| \tag{3.41}$$

which gives the mean distance for each contour point $\mathbf{p}_k^i$ of V^i to the closest one in V^j. The normalization by $\|\mathbf{p}_k^i - \overline{\mathbf{c}}^i\|_2$, with $\overline{\mathbf{c}}^i$ the centroid of the silhouette of the object on V^j (the notation $\overline{\mathbf{c}}^i$ is chosen instead of $\overline{\mathbf{c}}^{V^i}$, for clarity reasons), is important for the online detection phase, for which a similar metric is used (see Section 4), in order to deal with

Frame 10 Clustering by Affinity Propagation [Frey 07].

The idea of Affinity Propagation is to cluster data based on an arbitrary similarity measure between pairs of data points, each cluster being represented by an actual data point, the cluster center, called the "exemplar". With the popular k-medoid techniques, an initial set of k exemplars is selected, which is iteratively refined, along with their corresponding cluster. However, the results can be sensitive to the initial selection, especially for a large number k of clusters. With Affinity propagation, each data point is initially considered as a potential "exemplar", with equal or non-equal likelihoods. These likelihoods are set by what is called "preferences" of the data points.
The goal is then to identify consistent "exemplars" and clusters by passing "messages" between data points. These "messages" are values which reflect the affinity that one data point has for choosing another data point as its "exemplar". They consist in two terms. The first one, called "responsibility" $r(i,k)$, is related to the probability of a data point k to be the "exemplar" of data point i, given the probabilities i chooses other points than k as exemplars. Symmetrically, the second term, called "availability" $a(i,k)$, is related to the probability that data point i chooses data point k as its "exemplar", given the probabilities k is the exemplar of other points than i.
They are recursively updated based on some energy function accounting for the similarity measures between pairs of data points and for the "messages" (availability, responsibility) with the other data points. Then, the value of k which maximizes $a(i,k) + b(i,k)$ sets points i as an exemplar, if $k = i$, or sets points k as the exemplar of point i if $k \neq i$. The updating procedure is terminated once the values of a and r, or the assignment of exemplars, remain quite constant.
An example, extracted from [Frey 07] can be seen on the Figure below. It shows the refinement of the determination of clusters and their exemplars.



Principle of Affinity propagation (from [Frey 07]). The more red or the less green a data point is, the more it can be set as a cluster center. The darkness of the arrow from i to k reflects the magnitude "message" that point i belongs to the exemplar k.

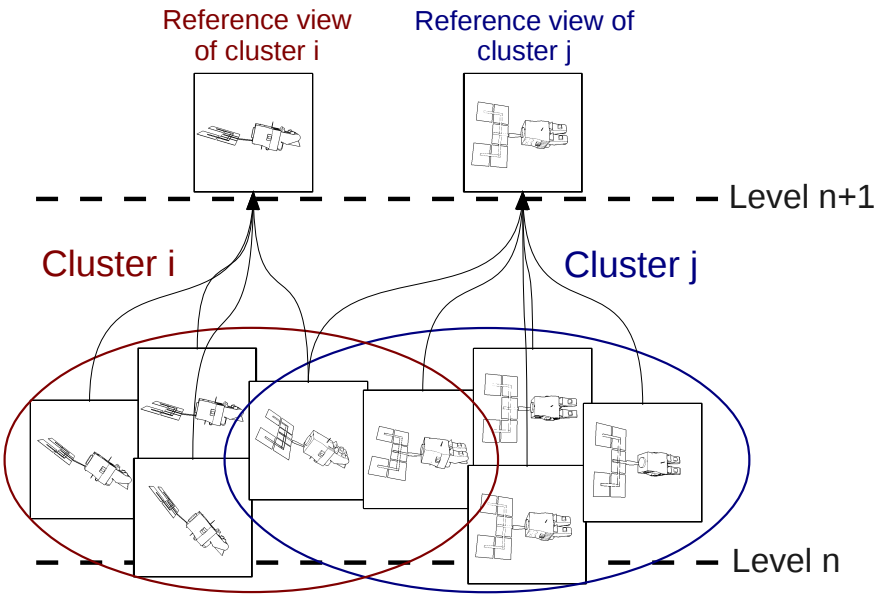


Figure 3.8 – Hierarchical clustering of synthetic views into a hierarchical view graph.

scale changes. $d_j(\mathbf{p}_k^i)$ can be computed fast by evaluating the distance transform of both views V^j.

This distance d_j^θ stands for the difference between the orientation $\theta(\mathbf{p}_k^i)$ of the contour point $\mathbf{p}_k^i$ and the orientation of the closest contour point $c_{arg(d_j(\mathbf{p}_k^i))^j}$ in V^j. Its weight is tuned by λ. It results in a discriminative similarity metric by taking into account distance between contours of the views and the difference between their corresponding orientations.

3.3 Matching and aligning synthetic views with images: a probabilistic approach

Our problem consists in matching and aligning the reference model views $\{V^j\}_{j=1}^{N_r}$ to each input image and finding the most likely one. Once a first image is segmented, at time step k_0, the next input images are used to determine a pose estimate. In order to ensure smooth transitions between the aligned model views, we propose a probabilistic framework to determine the best view.

Let us first describe how the pose can be determined from the alignment of a given synthetic view V with an input image $\mathbf{I}$.

3.3.1 Rough pose computation assuming a weak perspective model

We assume a weak perspective projection model, justified in our applications by the fact that the dimensions of the target space object are small relatively to the distance from the camera. Based on this assumption, the pose ${}^c\mathbf{M}_o$ between the camera and the object can be retrieved using the stored pose ${}^c\mathbf{M}_o^V$ used to generate the considered synthetic view V and the similarity transformation which aligns the view V with the image $\mathbf{I}$. This similarity transformation can be represented by four parameters: the in-plane rotation, expressed by a rotation angle β, the 2D translation vector $\mathbf{t} = \begin{bmatrix} t_x & t_y \end{bmatrix}^T$, and the scaling s. Let $\mathbf{R}_{-\beta}$ denotes the 3D rotation matrix of angle $-\beta$ around the optical axis $\mathbf{z}_c$. The rotation matrix ${}^c\mathbf{R}_o$ of ${}^c\mathbf{M}_o$ can then be computed as follows:

$$ {}^c\mathbf{R}_o = \mathbf{R}_{-\beta}\ {}^c\mathbf{R}_o^V \tag{3.42} $$

with ${}^c\mathbf{R}_o^V$ the rotation matrix of ${}^c\mathbf{M}_o^V$. Let us denote ${}^c\mathbf{t}_o^V = \begin{bmatrix} t_X^V & t_Y^V & t_Z^V \end{bmatrix}^T$ the translation vector of ${}^c\mathbf{M}_o^V$. Since scaling is assumed isotropic, we have $t_Z = s\ t_Z^V$ and the translation vector ${}^c\mathbf{t}_o$ of $\mathbf{M}$ is given by:

$$ {}^c\mathbf{t}_o \quad = \quad \begin{bmatrix} t_X^V + t_Z^V\ t_x \\ t_Y^V + t_Z^V\ t_y \\ s\ t_Z^V \end{bmatrix}. \tag{3.43} $$

However, since synthetic views are generated on a view sphere centered on the 3D model, at a distance d_0 (section 3.2), ${}^c\mathbf{t}_o^V = \begin{bmatrix} t_X^V & t_Y^V & t_Z^V \end{bmatrix}^T = \begin{bmatrix} 0 & 0 & d_0 \end{bmatrix}^T$. ${}^c\mathbf{t}_o$ thus becomes:

$$ {}^c\mathbf{t}_o \quad = \quad t_Z \begin{bmatrix} t_x \\ t_y \\ 1 \end{bmatrix}. \tag{3.44} $$

Finally, the pose ${}^c\mathbf{M}_o$ can be built based on ${}^c\mathbf{R}_o$ and ${}^c\mathbf{t}_o$ (equation (2.3)).

In the next sections, in order to determine a consistent pose ${}^c\mathbf{M}_o$, we present our solution to align each reference view V_r^j with the input images (section 3.3.2) and to determine the best matching (or most likely) reference view (section 3.3.4).

3.3.2 Aligning a reference view by refining similarity transformation parameters

Using the segmentation technique presented in section 3.1, the silhouette of the object can be extracted on the first segmented frame $\mathbf{I}_{k_0}$. The centroid $\overline{\mathbf{c}} = \begin{bmatrix} \overline{u} & \overline{v} \end{bmatrix}$, orientation α and area A of this silhouette can be then evaluated, using the image moments, as done in section 3.2.1. Figure 3.9 illustrates these steps.

Given a reference view V^j and using its stored silhouette parameters $\begin{bmatrix} \overline{u}^j & \overline{v}^j & \alpha^j & A^j \end{bmatrix}^T$ (to simplify notations, instead of $\begin{bmatrix} \overline{u}^{V^j} & \overline{v}^{V^j} & \alpha^{V^j} & A^{V^j} \end{bmatrix}^T$), we can retrieve the similarity transformation to align V^j with $\mathbf{I}_{k_0}$. This similarity transformation can be expressed by vector $\mathbf{x}$:

$$ \mathbf{x}^j = \begin{bmatrix} t_u \\ t_v \\ \beta \\ s \end{bmatrix} = \begin{bmatrix} \overline{u} - \overline{u}^j \\ \overline{v} - \overline{v}^j \\ \alpha - \alpha^j \\ \sqrt{\frac{A}{A^j}}. \end{bmatrix} \tag{3.45} $$

Figure 3.9 – Foreground/background segmentation (middle) of the input image (left), and computation of the centroid $\bar{\mathbf{c}}$, orientation α and area A of the extracted silhouette.

Based on the equations derived in section 3.3.1, the pose ${}^{c}\mathbf{M}_{o}^{j}$ (used to generate V^j) and $\mathbf{x}^j$ can provide us with a pose ${}^{c}\mathbf{M}_{o}$, for the considered view V^j. However, due to some segmentation errors, $\begin{bmatrix}\overline{u} & \overline{v} & \alpha & A\end{bmatrix}^T$ may be too coarsely computed.

We thus propose, for each reference view V^j, to refine parameters $\mathbf{x}^j$.

With the aim of estimating and refining $\mathbf{x}^j$ by minimizing a an error function or similarity measure between a reference view and the observed input image, different solutions can be investigated. Using a coarse-to-fine search around would provide an accurate estimate but is not computationally optimal. A local non-linear deterministic minimization technique such as Gauss-Newton or Levenberg-Marquardt, as proposed in the next chapter, could also be considered. However, such methods are too sensitive to local minima, especially with a coarse initialization as the one provided by the segmentation and the relative consistency of the treated reference view with respect to the image. Beside, the fast and robust similarity measure we propose is highly non-linear, resulting in cumbersome Jacobian computations. The robust and convenient solution we propose to rely on particle filtering, which is particularly suited for non-linear problems.

3.3.3 Refining as particle filtering

Given a reference model view V^j and a first image $\mathbf{I}_{k_0}$, we estimate and refine the corresponding $\mathbf{x}^j$ using particle filtering. The principle of particle filtering is recalled in Frame 11, and we propose to use the CONDENSATION [Isard 98] formulation of the filter, whose steps are recalled hereafter, and which illustrated on Figure 3.10.

In this sense $\mathbf{x}_k^j$ is represented by a finite set $\{\mathbf{x}_k^{(i,j)}\}_{i=1}^{N_j}$ of N_j samples, or particles, associated with weights $\{w_k^{(i,j)}\}_{i=1}^{N_j}$, with $\sum_{i=1}^{N_j} w_k^{(i,j)} = 1$. Then the process consists in the following classical steps:

1. **Initialization**: we set $\mathbf{x}_{k_0}^j = \mathbf{x}_{k_0}$ with $\mathbf{x}_{k_0}$ corresponding to the parameters evaluated from the first segmented frame $\mathbf{I}_{k_0}$.

2. **Evolution**: the particles $\{(\mathbf{x}_{k-1}{}^{(i,j)}, \frac{1}{N_j})\}_{i=1}^{N_j}$ are propagated according to a motion model, giving a new set: $\{(\mathbf{x}'_k{}^{(i,j)}, \frac{1}{N_j})\}_{i=1}^{N_j}$. We have considered a simple Gaussian

noise, so that:

$$\mathbf{x}_k^j = \mathbf{x}_{k-1}^j + \mathbf{v}_{k-1}^j \tag{3.46}$$

with $\mathbf{v}_k^j \sim \mathcal{N}(0, \mathbf{Q}_k)$.

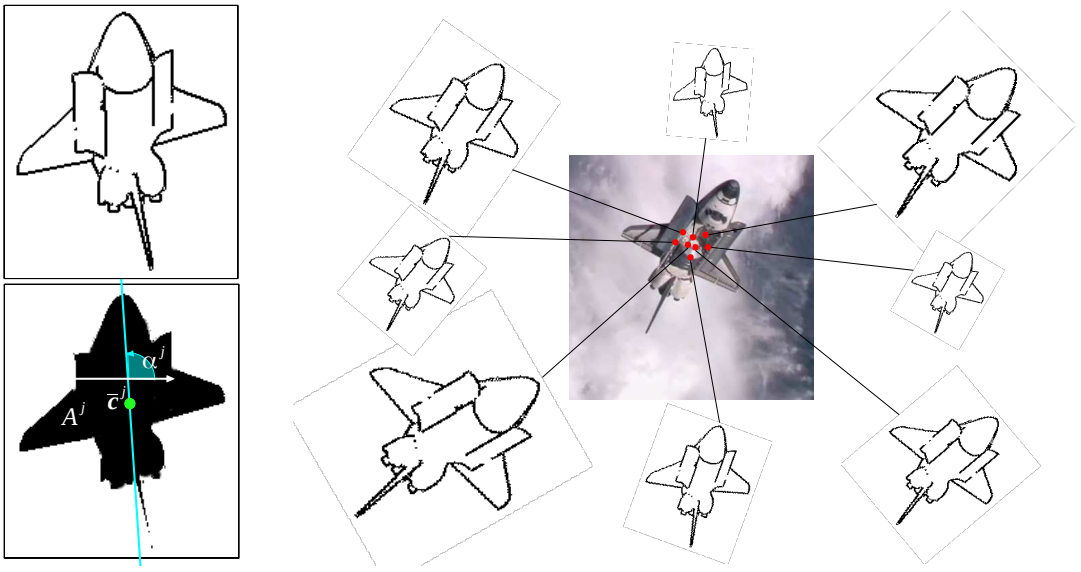
3. **Update**: the weight

$$w_k^{(i,j)} \propto p(I_k \mid \mathbf{x}_k^j = \mathbf{x}_k^{(i,j)} \tag{3.47}$$

of each predicted particle is computed by the likelihood function defined in Section 4.3. It provides a new set $\{(\mathbf{x}'_k{}^{(i,j)}, w_k^{(i,j)})\}_{i=1..N_j}$ with $\sum_{i=1}^{N_j} w_k^{(i,j)} = 1$.

4. **Random weighted draw**: random weighted draw of the particles $\{(\mathbf{x}'_k{}^{(i,j)}, w_k^{(i,j)})\}_{i=1}^{N_j}$ is performed, giving the set: $\{(\mathbf{x}_k{}^{(i,j)}, \frac{1}{N_j})\}_{i=1}^{N_j}$. The particle filtering output considered is the estimator of probability expectation:

$$\widehat{\mathbf{x}}_k^j = E[\mathbf{x}_k^j] = \sum_{i=1}^{N} \mathbf{x}_k^{(i,j)}. \tag{3.48}$$



(a) (b)

Figure 3.10 – Particle filtering for a reference view V^j (a). On (b), A particle i corresponds to the reference view which is translated, rotated and scaled with respect to $\mathbf{x}^{(i,j)}$ and the likelihood is evaluated using a similarity measure with the input image (b).

Likelihood evaluation

The likelihood function needs to be evaluated for each particle to compute their weight. The function chosen here is derived from a similarity measure similar to the one presented in section 3.2.3. For a model view V^j, a particle $\mathbf{x}_k^{(i,j)}$, an image $\mathbf{I}_k$ and its corresponding

Frame 11 Particle filtering

With the goal of approximating the general Bayesian filter (see Frame 5), particle filtering is based on the particle approximation which is derived from the Monte-Carlo theory. The idea of particle approximation is to represent a probability density function $p(\mathbf{x})$, such as the ones expressed in equations (2.46) and (2.47) by a set of N independent and identically distributed samples or particles $\{x^{(i)}\}_{i=1}^{N}$. This can be justified by the fact that the expectation over the samples is an estimator of the general formulation, and this approximation can be written as:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \simeq p^N(\mathbf{x}) = \sum_{i=1}^{N} \pi^{(i)} \delta_{x^{(i)}}(\mathbf{x}) \tag{3.49}$$

with $\pi^{(i)} \geq 0$ being the weights of particles $x^{(i)}$, with $\sum_{i=1}^{N} \pi^{(i)} = 1$.
Particle filters are thus based on this approximation, where particles represent a hypotheses of a state $\mathbf{x}$ of a considered system. The particle approximation is used to express the Bayesian filter prediction and correction equations (2.46) and (2.46), given the approximation of equation (3.49) at time step $k-1$:

$$p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1}) \simeq p^N(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1}) = \sum_{i=1}^{N} \pi_{k-1}^{(i)} \delta_{x_{k-1}^{(i)}}(\mathbf{x}_{k-1}). \tag{3.50}$$

For the **prediction** step, it is obtained through:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) \simeq p^N(\mathbf{x}_k \mid \mathbf{z}_{1:k-1}) = \sum_{i=1}^{N} \pi_{k-1}^{(i)} \delta_{x_k^{(i)}}(\mathbf{x}_k) \tag{3.51}$$

where particles $x_k^{(i)}$ are sampled with respect to $p(\mathbf{x}_k \mid \mathbf{x}_{1:k-1} = x_{k-1}^{(i)})$, which corresponds to the assumed dynamic or evolution model of the system, giving a new set of particles representing the a priori density $p(\mathbf{x}_{k-1} \mid \mathbf{z}_{1:k-1})$.
And the **correction** step is given by:

$$p(\mathbf{x}_k \mid \mathbf{z}_{1:k}) \approx p(\mathbf{z}_k \mid \mathbf{x}_k) \sum_{i=1}^{N} \pi_k^{(i)} p(\mathbf{x}_k \mid \mathbf{x}_{k-1}^{(i)}) \tag{3.52}$$

with weights being computed as:

$$\pi_k^{(i)} = \frac{\pi_{k-1}^{(i)} p(\mathbf{z}_k \mid \mathbf{x}_k = x_k^{(i)})}{\sum_{i=1}^{N} \pi_{k-1}^{(i)} p(\mathbf{z}_k \mid \mathbf{x}_k = x_k^{(i)})}. \tag{3.53}$$

$p(\mathbf{z}_k \mid \mathbf{x}_k = x_k^{(i)})$ are the likelihoods of particles $x_k^{(i)}$ which are computed using an observation. This weighting process of the new set of particles thus represents the a posteriori density $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$, and the a posteriori estimate state of the system is given by:

$$\widehat{\mathbf{x}_k} = \sum_{i=1}^{N} \pi_k^{(i)} x_k^{(i)} \tag{3.54}$$

This formulation is used by the SIS algorithm, for which particle are initialized at a particular state and weights are initially set to $\frac{1}{N}$.
However this algorithm can suffer from degeneracy since most of particles tends to be assigned law weights. In order to avoid this phenomenon, a resampling process can be carried out by discarding particles with low weights. The SIR (for Sampling Importance Resampling) and CONDENSATION [Isard 98] algorithms have been designed in that sense. They consist in performing a random weighted draw providing a restricted set of the most likely particles, and in duplicating them to keep a constant number N of particles.

segmented image $\mathbf{I}_k^{seg}$, it consists in the distance $D(\mathbf{x}_k^{(i,j)})$ between the contour points $\{\mathbf{p}_l^{i,j}\}_{l=1}^{M_\mathbf{x}}$ extracted from V^j translated, scaled and rotated around its centroid with respect to $\mathbf{x}_k^{(i,j)}$, and the corresponding closest contour points of both sets $\{\mathbf{p}_{l,k}\}_{l=1}^{N}$ and $\{\mathbf{p}_{l,k}^{seg}\}_{l=1}^{P}$ extracted from $\mathbf{I}_k$ and $\mathbf{I}_k^{seg}$ using a Canny edge detector:

$$D(\mathbf{x}_k^{(i,j)}) = \rho d(\mathbf{x}_k^{(i,j)}, \mathbf{I}_k) + \rho_{seg} d(\mathbf{x}_k^{(i,j)}, \mathbf{I}_k^{seg}) \tag{3.55}$$

$$\text{with} \quad d(\mathbf{x}_k^{(i,j)}, \mathbf{I}_k) = d_I + \lambda d_I \tag{3.56}$$

$$= \frac{1}{M_\mathbf{x}} \sum_{i=0}^{M_\mathbf{x}} (d_I(\mathbf{p}_i^{i,j}) + \lambda d_I^\theta(\mathbf{p}_i^{i,j})) \tag{3.57}$$

ρ and ρ_{seg} are constant weights, tuning the balance between the original image and the segmented one. $d_I(\mathbf{p}_i^\mathbf{x})$ and $d_I^\theta(\mathbf{p}_k^\mathbf{x})$ are respectively computed in similar ways to (3.40) and (3.41). Assuming a Gaussian distribution of similarity measure D, the likelihood $\pi_k^{(i,j)}$ of $\mathbf{x}_k^{(i,j)}$ for a frame $\mathbf{I}_k$, with τ a tuning parameter, is given by:

$$w_k^{(i,j)} \propto \pi_k^{(i,j)} = e^{-\tau^{-1} D(\mathbf{x}_k^{(i,j)})^2} \tag{3.58}$$

3.3.4 Matching the reference views within a probabilistic framework

Once the particle filtering is performed for all the reference views V^j for a frame $\mathbf{I}_k$, the goal is then to find the most likely view, while ensuring smooth transitions with respect to previous selected views. For this purpose, probabilistic graphical models can be considered. We have chosen to employ Hidden Markov Models (HMM) [Rabiner 89] which define a joint distribution over the successive selected model views.

The sequence of the matched views as a Hidden Markov Model

A HMM models a sequence of observations overlying a sequence of "hidden" (not directly observables) states. In our case, the sequence of the states is the sequence of the matched reference views $\{V_l\}_{l=k_0}^k$, with $V_l \in \{V^j\}_{j=1}^{N_r}$. The sequence of observations are the images $\{\mathbf{I}_l\}_{l=k_0}^k$, or more precisely the output of the particle filtering step performed for each reference view which are the weights $\pi_l^{(i,j)}$ of the particles and the estimate $\widehat{\mathbf{x}}_k^j$, for $\{V^j\}_{j=1}^{N_r}$, given $\mathbf{I}_k$. A HMM supposes that $\{V_l\}_{l=k_0}^k$ follows a hidden Markov process, meaning that each state, or each matched reference view V_k at time step k, depends only on the previous state, or on the previous matched reference view V_{k-1}, regardless $V_{k-2}, V_{k-3} \cdots$. Besides, each observation $\mathbf{I}_l$ is assumed to only depend on V_l. Based on these assumptions, the joint probability of the sequence $\{V_l\}_{l=k_0}^k$ and the sequence $\{\mathbf{I}_l\}_{l=k_0}^k$ can be written as:

$$p(V_{k_0:k}, \mathbf{I}_{k_0:k}) = \prod_{l=k_0}^{k} p(\mathbf{I}_l|V_l) p(V_l, V_{l-1}) \tag{3.59}$$

The probability $p(\mathbf{I}_l|V_l)$ refers to the observation probability of a given matched model view V_l. If V^j is the view corresponding to V_l in $\{V^j\}_{j=1}^{N_r}$, then:

$$p(\mathbf{I}_l|V^j) = \frac{1}{A}\sum_{i=1}^{N_j}\pi_l^{(i,j)} \tag{3.60}$$

where $\pi_l^{(i,j)}$ is the weight of particle $\mathbf{x}_l^{(i,j)}$ and A is a normalization factor so that we deal with a probability distribution:

$$A = \sum_{j=1}^{N_r}\sum_{i=1}^{N_j}\pi_l^{(i,j)}. \tag{3.61}$$

Equation (3.60) means that the sum of the weight of the particles of a reference view is chosen as the global weight of the considered reference view.

$p(V_l, V_{l-1})$ is the transition probability between matched views V_l and V_{l-1}. It can be defined offline by:

$$p(V_l, V_{l-1}) \propto e^{-\frac{acos(\mathbf{u}_l{}^T\mathbf{u}_{l-1})^2}{2{\sigma_v}^2}} \tag{3.62}$$

where $\mathbf{u}_l$ corresponds to the viewpoint vector of the matched view V_l in the set $\{V^j\}_{j=1}^{N_r}$. This viewpoint vector is related to the azimuth and elevation angles used to generate the synthetic view corresponding to V_l. The computation of $acos(\mathbf{u}_l{}^T\mathbf{u}_{l-1})$ gives the angle between $\mathbf{u}_l$ and $\mathbf{u}_{l-1}$, measuring the "distance" between the views corresponding to V_l and V_{l-1} on the viewsphere. The more distant are the views on the viewsphere, the less likely is the transition between them. σ_v a fixed parameter related to the variance of the viewpoints.

Inference of the HMM

In order to infer this HMM, what consists in maximizing equation (3.59) with respect to the sequence of views, in the set $\{V^j\}_{j=1}^{N_r}$ at time step k and thus to determine V_k (the last element of the estimated sequence), we use the classical Viterbi algorithm [Rabiner 89]. The resulting reference view V^{j^*} corresponding to V_k is thus chosen as the most likely one.

Global estimate of the similarity transformation parameters

As an estimate of the similarity transformation parameters, we could have chosen the parameters $\widehat{\mathbf{x}}_k^{j^*}$ of V^{j^*}, resulting from the particle filtering step. However, we propose to consider the whole set of reference views to compute a global estimate $\widehat{\mathbf{x}}_k$, given their respective probabilities. It gives:

$$\widehat{\mathbf{x}}_k = \sum_{j=1}^{N_r} p(\mathbf{I}_k|V^j)\widehat{\mathbf{x}}_k^j \tag{3.63}$$

Rough estimate of the pose

Using V_k and $\widehat{\mathbf{x}}_k$ and based on steps presented in section 3.3.1 (note that t_u and t_v are first converted to metric coordinates to get t_x and t_y), a rough estimate of the pose ${}^c\mathbf{M}_o{}^k$ for $\mathbf{I}_k$ can be determined.

Let us note that the particles $\{(\mathbf{x}'^{(i,j)}_k\}_{i=1}^{N_j}$ of each V^j are finally reweighted with respect to $\widehat{\mathbf{x}}_k$, prior to being processed in the particle filters of the different reference views, for the next frame $\mathbf{I}_{k+1}$.

3.3.5 Pose refinement as graph search

Once a certain number of frames k_F is reached, the most likely reference model view V_{k_F}, serves as a starting point of a best match search among its child views on the hierarchical view graph and among the whole set of its associated particles.

More formally, if V^{j^*} denotes the reference view corresponding to V_{k_F}, the process results in a view $V^{l_{j^*}}$ determined at the bottom level on the view graph, and in a best particle $\widehat{\mathbf{x}}^{k_F}$:

$$\widehat{\mathbf{x}}_{k_F} = \mathbf{x}_{k_F}^{(\widehat{i}_F, \widehat{l}_{j^*})} \tag{3.64}$$

with

$$(\widehat{i}_F, \widehat{l}_{j^*}) = \underset{(i,l_{j^*})}{\arg\max}\, w_{k_F}^{(i,l_{j^*})}. \tag{3.65}$$

$w_{k_F}^{(i,l_{j^*})}$ are the weights of the particles of the view V^{j^*}, which are computed for the view $V^{l_{j^*}}$. With $\widehat{\mathbf{x}}_{k_F}$ and $V^{l_{j^*}}$ and using the steps in section 3.3.1, we can compute a refined pose ${}^c\mathbf{M}_o{}^{k_F}$. This pose is finally directly used to initialize a frame-by-frame tracking algorithm presented in chapter 4.

Having detailed the different steps of our detection initial pose estimation framework, we provide in the next section some experimental results to validate them.

3.4 Experimental results

The rendering process of the 3D polygonal model to generate synthetic views relies on OpenSceneGraph, which is flexible 3D rendering engine. Regarding hardware, a laptop with an NVIDIA NVS 3100M graphic card has been used, along with a 2.8GHz Intel Core i7 CPU. The algorithm has been run on synthetic images featuring a Spot satellite (512×512 images are processed). Concerning real sequences, a first one shows the Soyuz TMA-12 spacecraft approaching the International Space Station (ISS) (400×400 images). A second one features the Atlantis Space Shuttle performing its pitch maneuver towards the ISS (360×360 images). Finally, we have tested the method on a mock-up of the telecommunication satellite Amazonas (512×512 images). The 3D model of the Spot satellite has been provided by Astrium and the models for the Soyuz spacecraft and the shuttle can be found on Google 3D Warehouse, whereas the one of Amazonas has been designed manually since the provided version of model of the actual satellite differed significantly from the mock-up.

3.4.1 Results for the learning step

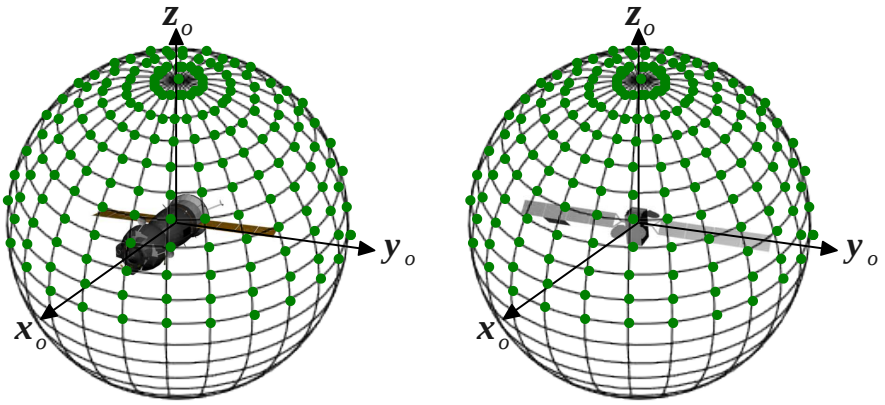
Table 3.1 shows the parametrization of the view sphere for the different objects, with sampling steps for both azimuth and elevation angles (in degrees). Let us note that for Soyuz, due to the symmetry of the object around axis $\mathbf{x}_o$ (Figure 3.11(a)), we have restricted

Objet	Azi. step	Elev. step	L0	L1	L2
Spot	8°	8°	2303	436	53
Atlantis	8.6°	8.6°	1765	304	44
Soyuz	5.1°	5.1°	1225	268	38
Amazonas	5.6°	5.6°	1025	259	20

Table 3.1 – Parametrization of the view sphere for each object and results of the learning step, with the number of reference determined at each level L of the hierarchical view graph.

to a half-sphere for the generation of the synthetic views. It has also been the case for Amazonas (see Figure 3.11(b)), due to the relative symmetry with respect to the plane (x, y).

Table 3.1 also presents the results of the building of the hierarchical model view graph and the number of reference views obtained at each level L of the graph. For each object, we have stopped the learning process at level 2 in order to get reasonable and usable numbers of reference views. Some examples of these reference views can be seen on Figure 3.12 and will be compared for the detection and pose estimation process (see section 3.4.4).



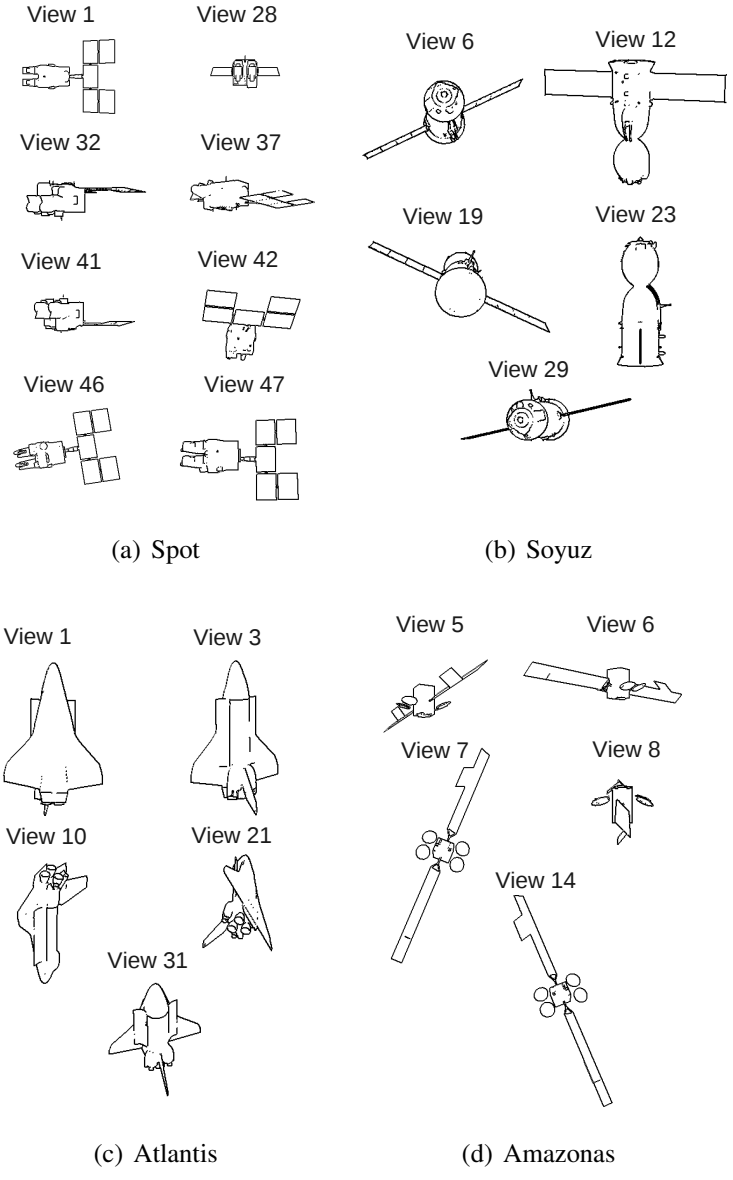
(a) Soyuz (b) Amazonas

Figure 3.11 – Generation of the synthetic views for the Soyuz (a) and Amazonas cases (b), for which half-spheres have been used.

3.4.2 Results for the foreground/background segmentation step

3.4.2.1 Terrestrial background

As already stated in the introduction of this chapter, the detection and initial pose estimation process starts, for its online phase, by segmenting the a set of initial images of the sequence. Figure 3.13 shows some sequences for which our segmentation method has been applied, in the case of a terrestrial background, which is the most challenging case. The first three rows represent three sequences featuring the Spot satellite and the next ones deal with the Soyuz and Atlantis spacecrafts.



(a) Spot (b) Soyuz

(c) Atlantis (d) Amazonas

Figure 3.12 – Some reference views determined at the second level of the view graph for Spot (a), Soyuz (b) and Atlantis (c).

The left column shows the Harris corner points being tracked using the KLT tracker (red trajectories) for the different sequences. Green and blue dots represent the dots respectively classified as belonging to the foreground object or the background, spread on regular grids. We observe that the clustering process explained in section 3.1.2.3 is performed correctly, with very few misclassified pixels for Spot and Soyuz whereas we find more errors for Atlantis.

Figures 3.13 on the middle column depict the mean image between the current image and its homography-based compensated one, enhancing the motion of the object with respect to the background, with the zone featuring the object being echoed and blurred (with $k_H = 5$, with k_H being defined in section 3.1.2.4). Finally, Figures 3.13 on the right show both object (colored) and the background (black) layers after the segmentation phase, which starts at $k_0 = 8$ (defined at the beginning of section 3.3), with satisfactory results, despite the cluttered background. For Atlantis, the errors observed on the background/object modeling are compensated by a good estimation of the background motion homography. The combination of background/foreground modeling through KLT points and motion compensation is also obvious for the Spot sequence featured on the third row. Some monochrome parts of the solar arrays are indeed not affected by the motion compensation, resulting in low likelihoods of being labeled as foreground pixels. However, with a good feature points classification, these likelihoods are increased thanks to a correct foreground modeling. This segmentation step for a frame is executed in $0.6s$ using kernel density estimation (section 3.18) and in $0.38s$ using histograms (section 3.23), on average for Spot sequence.

3.4.2.2 Deep space background

For this case, we have applied the simple technique presented in section 3.1.3 on a sequence featuring the Amazonas mock-up. Figure3.14 shows the input image and the resulting successive segmented ones. The result from a simple thresholding (using the same threshold value as the one to build the color histograms with our method) is depicted on Figure 3.14(e). We observe that our method rapidly converges to a correct and desirable segmentation, whereas thresholding provides a sparse and coarse segmentation for each frame.

3.4.3 Comparative study for the similarity measure

As a demonstration of the benefit of our similarity measure, we have compared it with the one used in [Steger 02, Ulrich 09] (introduced in section 2.4.1.1), which we refer a D_S, and from which the one proposed in [Hinterstoisser 10] is derived. As a reminder, D_S basically consists in the normalized dot products between the gradients of the template (or view here) and the underlying of the gradients of the image. In other words, it is the difference of gradient orientations (actually the cos of the difference) between the template and the image. It has proven its efficiency and robustness and in contrast to ours, [Steger 02] does not need any pior edge extraction.

In order to perform this comparative evaluation, we have selected, for the Atlantis example, the reference view V presented on Figure 3.15(a) (which is view 31 on Figure 3.12(a)) and which approximately matches with the test image shown on Figure 3.15(b).

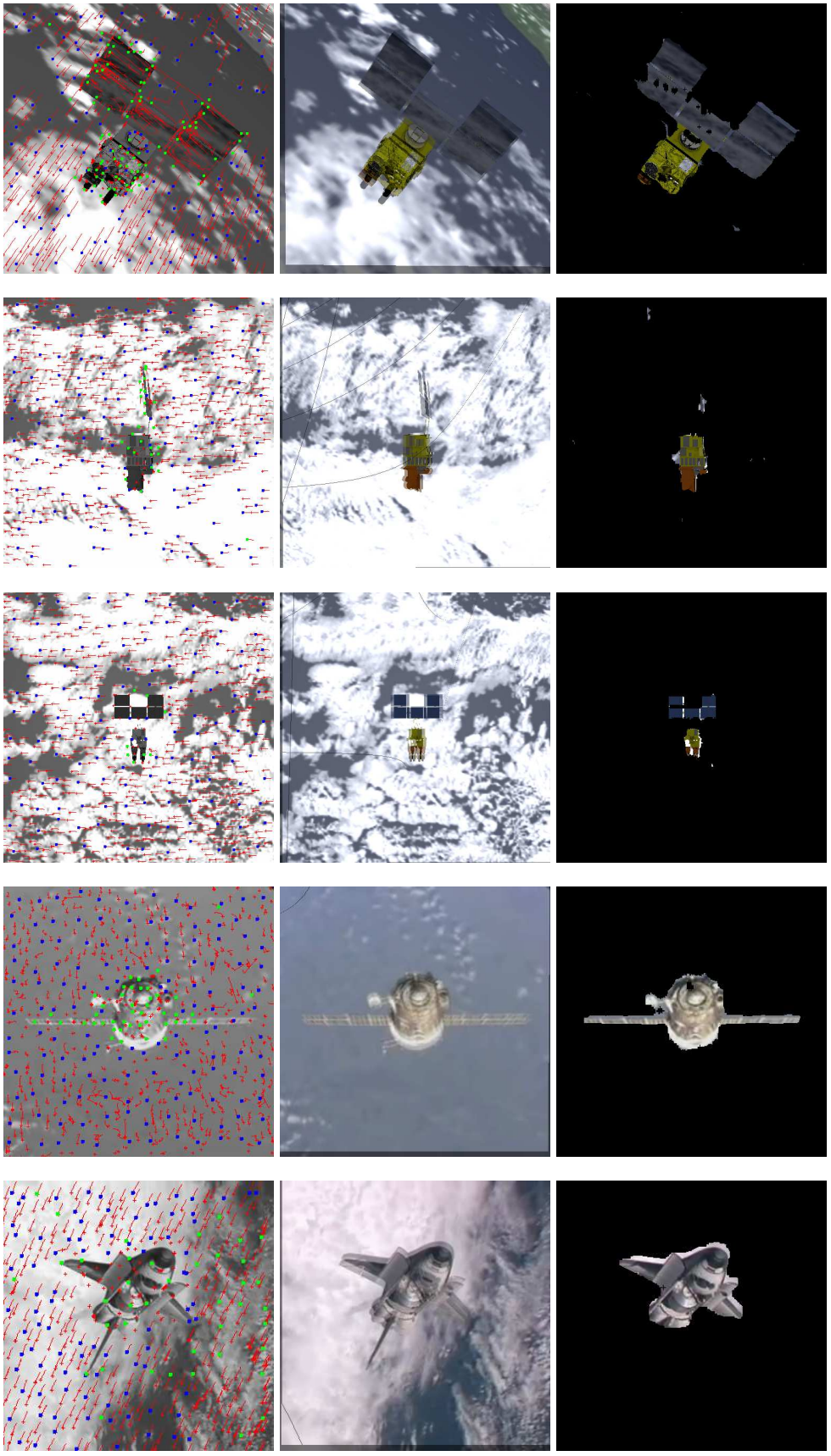
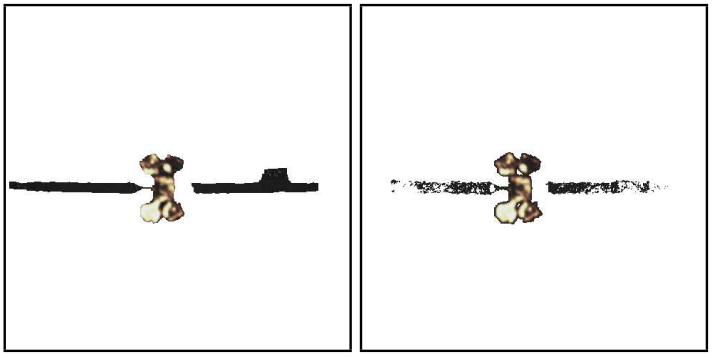


Figure 3.13 – Segmentation process for some sequences. On the left are shown the tracked KLT trajectory points, in green those classified as foreground points and in blue as background points. The middle column represents the mean image after homography based motion compensation and the right one the resulting segmented image.

(a) input frame 1 (b) segmented frame 1 (c) segmented frame 3



(d) segmented frame 8 (e) segmented frame 8 (thresholding)

Figure 3.14 – Segmentation process the Amazonas sequence. On (b,c,d) are represented successive segmented frames with our method and (e) depicts segmentation provided by a simple thresholding function.

The edge map of the test image (Figure 3.15(c)), obtained through a Canny edge detector, is also featured since it is processed for our similarity measure D. We have studied the behavior of both measures with respect to the similarity transform parameters

$$\mathbf{x} = \begin{bmatrix} t_u \\ t_v \\ \beta \\ s \end{bmatrix} = \begin{bmatrix} \overline{u} - \overline{u}^V \\ \overline{v} - \overline{v}^V \\ \alpha - \alpha^V \\ \sqrt{\frac{A}{A^V}} \end{bmatrix}^T \tag{3.66}$$

which are introduced in section 3.3.2, so that V is translated, rotated and scaled according to $\mathbf{x}$, on a regular discretization grid. A reference value $\mathbf{x}^*$ of these parameters is arbitrary set to a visually consistent value (regarding the image), and we have evaluated the similarity measures on regular grids over the plane (t_u, t_v) with the reference value (β^*, s^*) and conversely over the plane (β, s) with (t_u^*, t_v^*).

For our measure, we have actually tested d_I, d_I^θ and finally $D = d_I + \lambda\, d_I^\theta$. Results are represented on Figure 3.16 for our approach and on Figure 3.17 for D_S [Steger 02], along with the superimposition of the view V on the image at the resulting global minimum. We can observe that d_I (Figures 3.16(a),(b)) is smooth over the different parameters with a clear clear global minimum. It is however quite flat around this minimum and some local minima can be noticed when evaluating both (t_u, t_v) and (β, s).

With d_I^θ (Figures 3.16(c),(d)), the global minimum can be seen at the positions ($t_u \simeq -10, t_v \simeq -10$) on Figure 3.16(c) and it is sharp right around it over (t_u, t_v) and also for (β, s). The measure is otherwise rough, with many local minima. Over s, the curve is quite flat around the minimum.

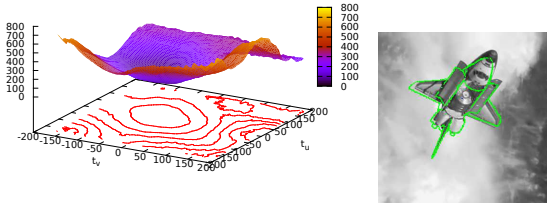
d_I and d_I^θ can thus be seen as complementary and we can observe on the plots of D (Figures 3.16(e),(f)) that combining them results in a sharper global minimum w.r.t. the different parameters, and the main local minima of both measures d_I and d_I^θ can be avoided, especially the ones over (β, s) for d_I.

In contrast, D_S, shows a very rough aspect, despite a global minimum is also found quite near the actual position. Since it is even rougher close to the border of the image when scanning parameters (t_u, t_v) for (β^*, s^*), the plots close to the borders have been removed for a clearer visualization. Through this rough aspect on this case, D_S appears to be more sensitive to some local minima.

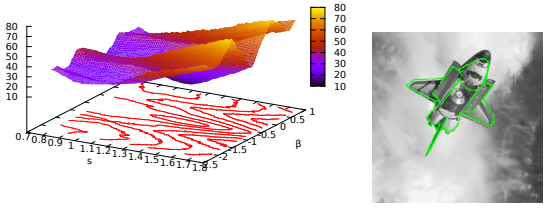
However, in this particular case, the edge map (Figure 3.15(c)) is quite noiseless, with few occlusions, making our approach particularly suitable. In the case of poor edge extraction in the original input image (in highly cluttered or occluded scenes), the advantage of our method with respect to D_S might thus be moderated since D_S, by directly dealing with the image gradients, does not require any edge extraction step. But as suggested by equation (3.56), our method has been made robust to highly cluttered by computing D on both the edge maps of the original image and the segmented image. Edges resulting from a potentially cluttered background (for instance on Figure 3.13(a)) can indeed be removed with the segmented image, whereas potential segmentation errors can be compensated by keeping the original image in the computation of D.

(a) (b) (c)

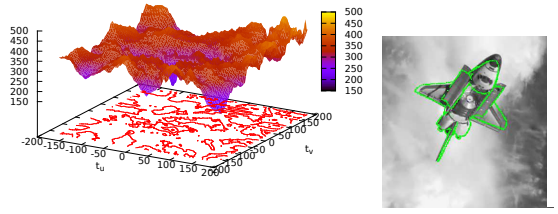
Figure 3.15 – Chosen reference view of the Atlantis shuttle (a), test image (b) and edge map of the test image (c).



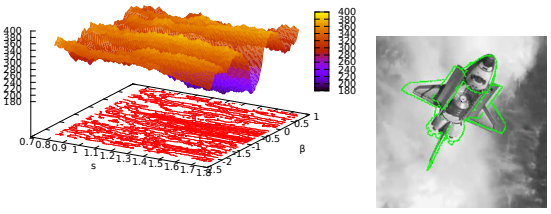
(a) Similarity measure d_I w.r.t to (t_u, t_v), at the reference position (β^*, s^*)



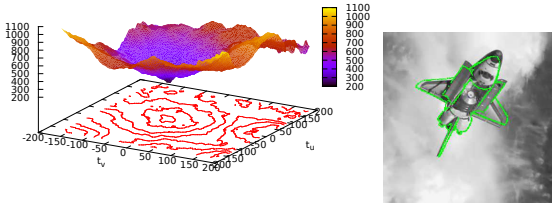
(b) Similarity measure for d_I w.r.t to (β, s) at the reference position (t_u^*, t_v^*)



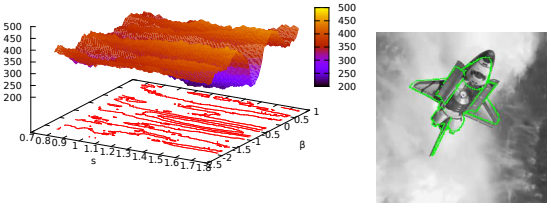
(c) Similarity measure d_I^θ w.r.t to (t_u, t_v), at the reference position (β^*, s^*)



(d) Similarity measure for d_I^θ w.r.t to (β, s) at the reference position (t_u^*, t_u^*)

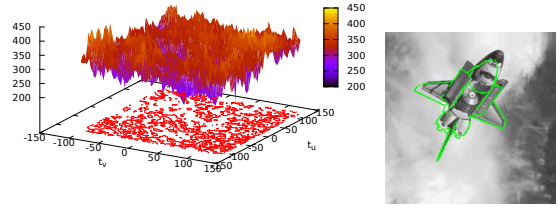


(e) Similarity measure d_I^θ w.r.t to (t_u, t_v), at the reference position (θ^*, s^*)

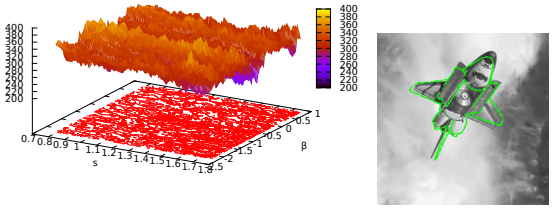


(f) Similarity measure for d_I^θ w.r.t to (β, s) at the reference position (t_u^*, t_u^*)

Figure 3.16 – Similarity measures for d_I (top), d_I^θ (middle) and D (bottom) with respect to (t_u, t_v) (left) and (β, s) (right), along with the superimposition of the reference view at the determined global minimum.



(a) Similarity measure for D_S w.r.t to (t_u, t_v), at the reference position (β^*, s^*)



(b) Similarity measure for D_S w.r.t to (β, s) at the reference position (t_u^*, t_v^*)

Figure 3.17 – Similarity measure for D_S

3.4.4 Results for the initial pose estimation

Several sequences, with some of them presented on Figure 3.13, have been processed within our detection and pose estimation framework. For the four considered objects, reference views collected at the second level (L2) of their respective hierarchical view graph (see Figure 3.12) are selected to perform the matching and alignment phase described in section 3.3, which is done over 10 initial input frames. The particle filters on the similarity transformation parameters, described in section 3.3.3 are built of 100 particles in these tests.

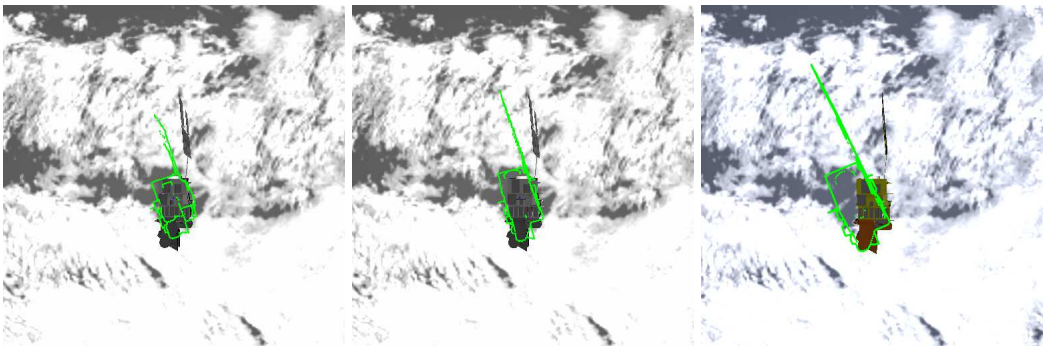
Results for the different sequences are depicted on Figures 3.19- 3.27. The initial segmented frame is shown on (a). The probabilistic alignment phase is represented by the superimposition of the most likely reference view, on different input frames. The pose refinement step, using a best match search through the hierarchical view graph, starting from the most likely reference view at frame 9, is shown. The projection of the 3D model with respect to the pose estimated by the frame-by-frame model-based tracking algorithm (chapter 4), used, the pose provided by our detection system is also featured.

In order to show the advantage the Hidden Markov Model used to smoothly match the reference views with the input image (section 3.3.4), both observation and marginal joint probabilities of the different reference views along the input sequence, are plotted. The observation probabilities correspond to equation (3.60) and are related to the likelihood evaluation of the particles of the considered views (section 3.3.3). Marginal joint probabilities of the views are provided by the inference of the HMM through the Viterbi algorithm [Rabiner 89]. As an inference method, the Viterbi algorithm indeed aims at determining the state sequence (the sequence of views) which maximizes the global (over the whole sequence) joint probability given by equation (3.59). Through this algorithm, marginal joint probabilities of each element of the determined sequence can be given, and here is represented the probability of the last element of the sequence, which the most likely view V_k at a given time step k.

For sequences on Figure 3.19, 3.20, 3.22, and 3.26, we observe that consistent reference views are matched and realigned to the image through particle filtering. The benefit of the HMM is visible through its ability to *smooth*, by estimating the optimal sequence, the determination of the most likely view at each time step. It is particularly the case the Spot sequence 3 (Figure 3.21) for which three reference views (views 1, 46 and 47)

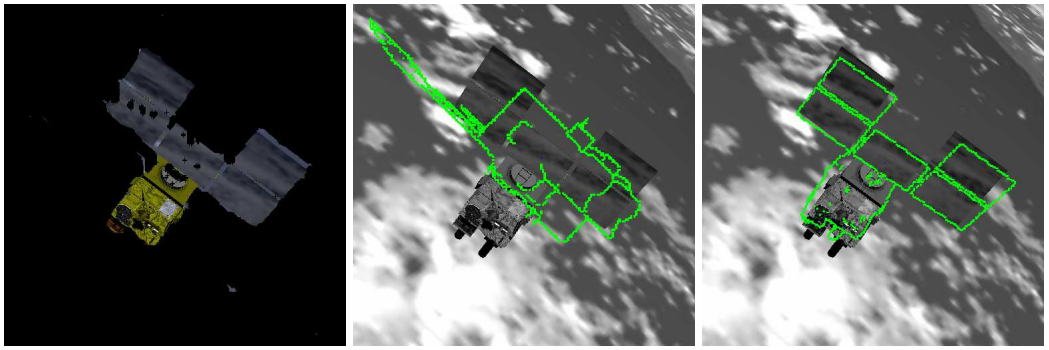
still have similar appearance, despite the hierarchical clustering technique described in section 3.2. Views 46 and 47 have been generated from quite opposite viewpoints with respect to view 1. We observe that their observation probabilities tend to be similar, with some switches and with view1 being initially preponderant. Since view 46, which is a consistent match, has a slightly larger observation probability than view 1 and since view 47 (which is spatially close to view 46) is gaining likelihood along the sequence, view 1 is progressively rejected, in terms of marginal joint probability, thanks to the inference of the HMM, and the marginal probability of view 46 is increasing. False positive can also be observed on the initial match for the Spot sequences 1 (Figure 3.19) and 5 (Figure 3.23), for instance due to the coarse segmentation for sequence 5 (Figure 3.23(a)). Likelihoods of actual matches remaining large or increasing through the particle filtering refinement, the HMM rapidly discard these false positives and probabilities of actual matches reach large values with respect to the other reference views. Similar observations can be made for the Amazonas sequence on Figure 3.24, with the two quite similar views (7 and 14), through the progressive refinement and, the HMM disambiguates. Ambiguities can also be observed for the Soyuz sequence 3.25 between view 6 and view 19.

On Figure 3.20, due to the coarse segmentation, the orientation of the object in the image is initially not proper, as seen on Figure 3.20(b), but through the particle filtering framework, the most likely view, which is consistent with the image, is progressively aligned, its marginal probability increasing. On this sequence we can see the effect of removing the particle filtering refinement step by directly traversing through the view graph at the first frame, from the most likely view using $\mathbf{x}^j = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$, and initializing the frame-by-frame tracking (see Figure 3.18). Despite a consistent view can be determined, tracking fails.

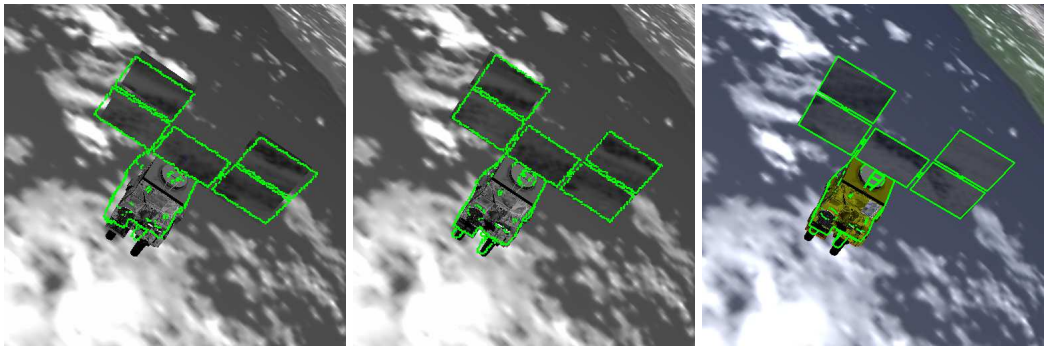


(a) Matched view 0 (b) Pose refinement (c) Initialized tracking

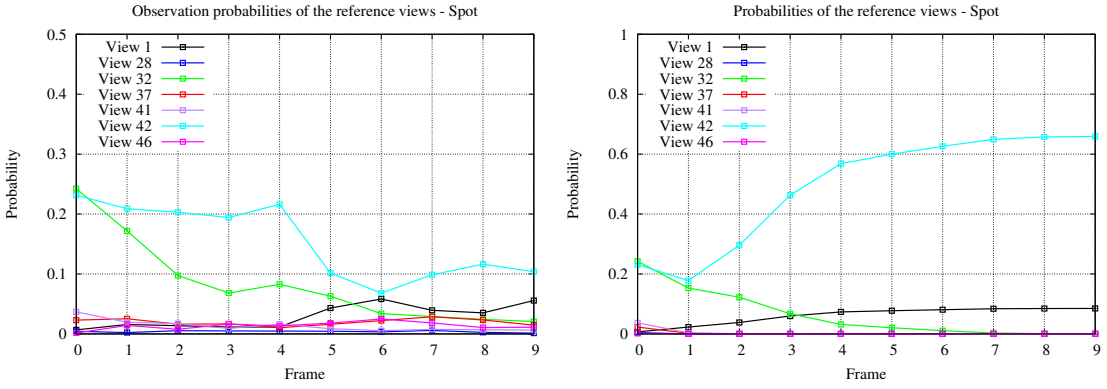
Figure 3.18 – Segmentation, detection and pose estimation process - Spot sequence 1



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 2

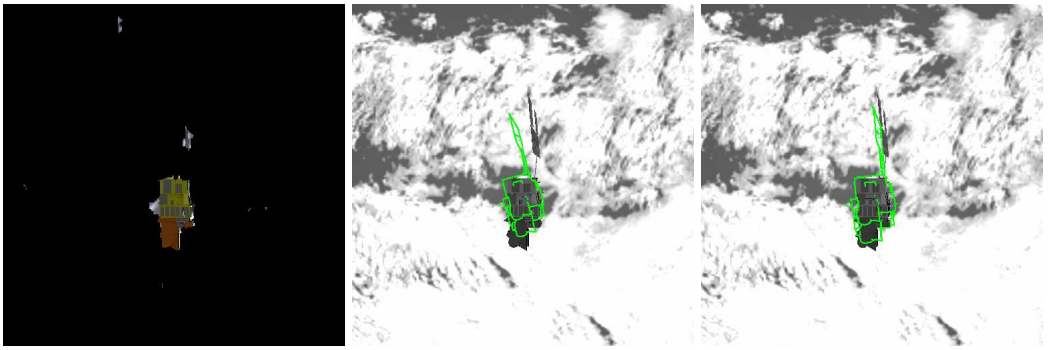


(d) Aligned view 9 (e) Pose refinement (f) Initialized tracking

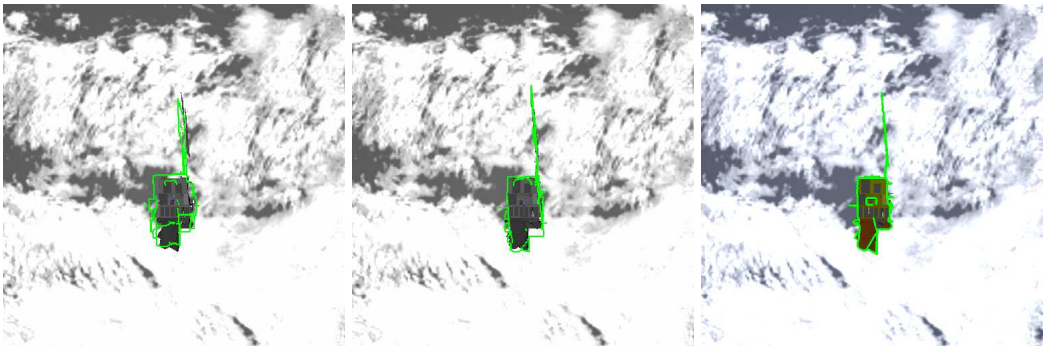


(g) Observation probabilities of the views (h) Marginal joint probabilities of the views

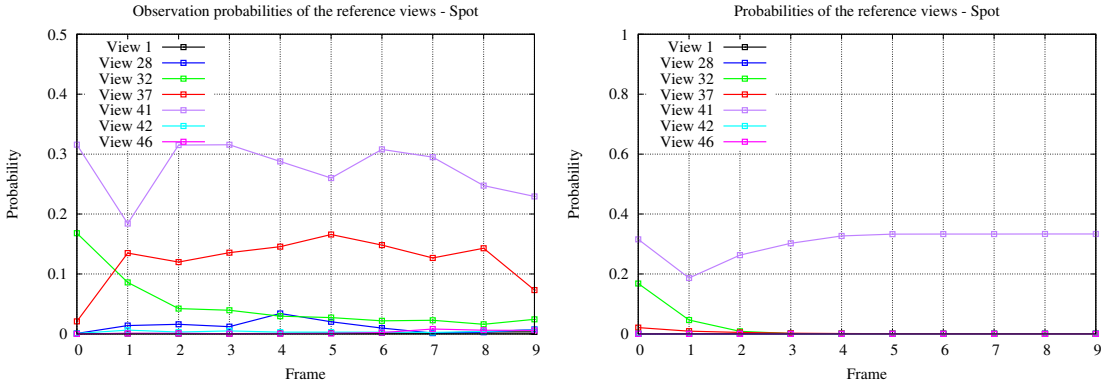
Figure 3.19 – Segmentation, detection and pose estimation process - Spot sequence 1



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 2

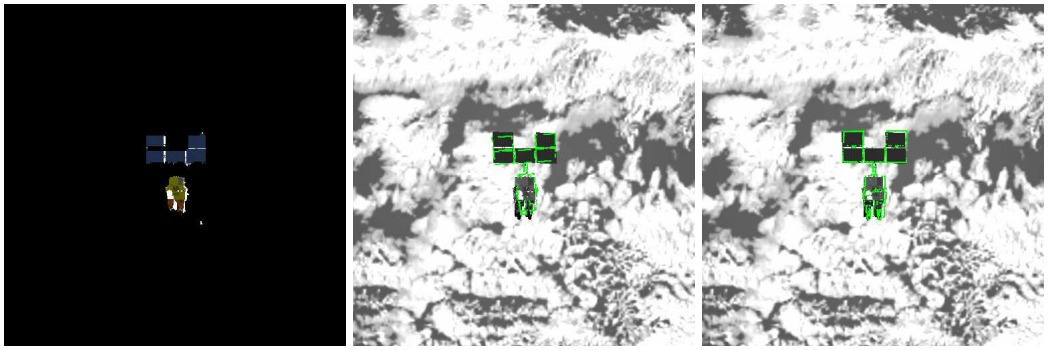


(d) Aligned view 9 (e) Pose refinement (f) Initialized tracking

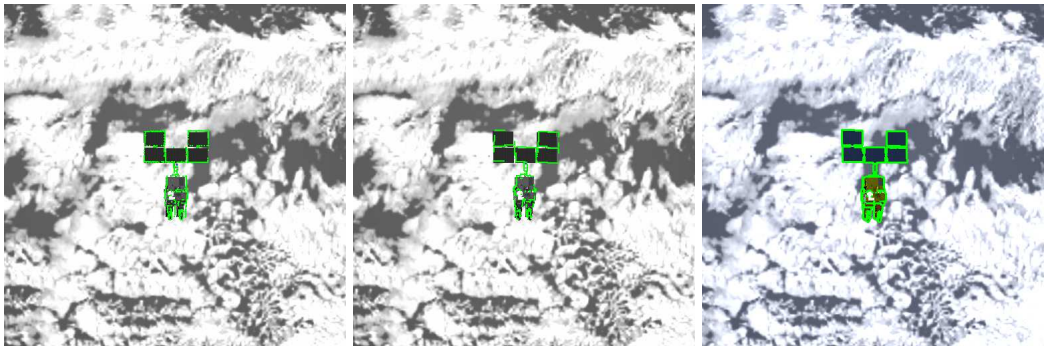


(g) Observation probabilities of the views (h) Marginal joint probabilities of the views

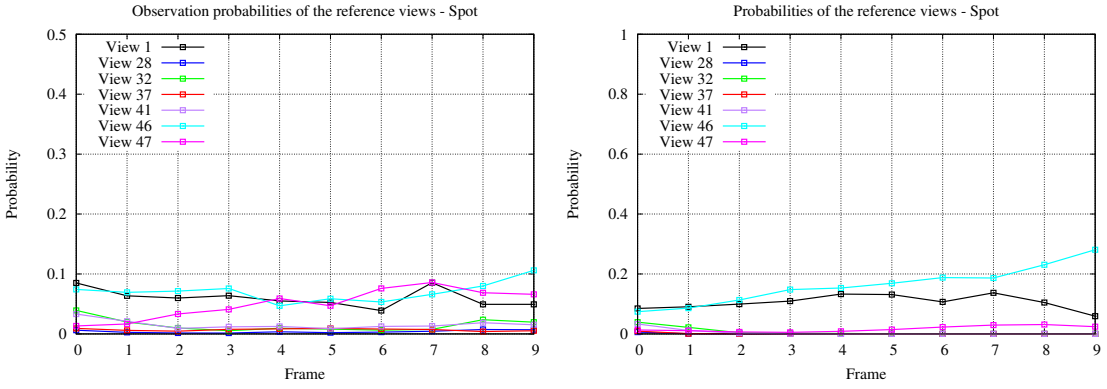
Figure 3.20 – Segmentation, detection and pose estimation process - Spot sequence 2



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 2

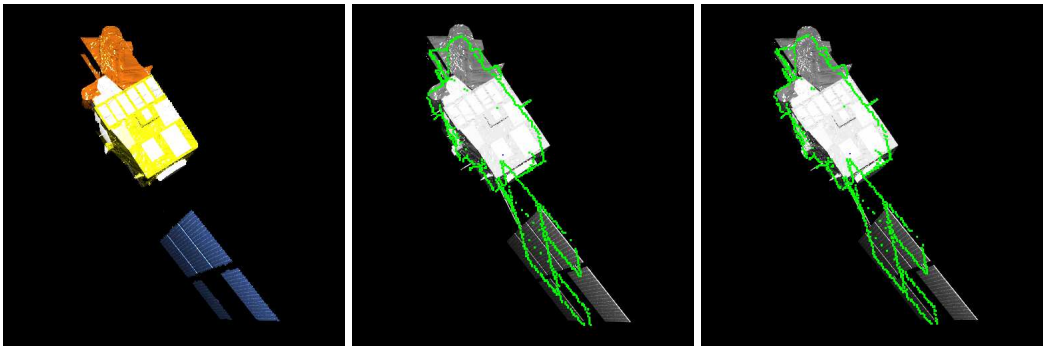


(d) Aligned view 9 (e) Pose refinement (f) Initialized tracking

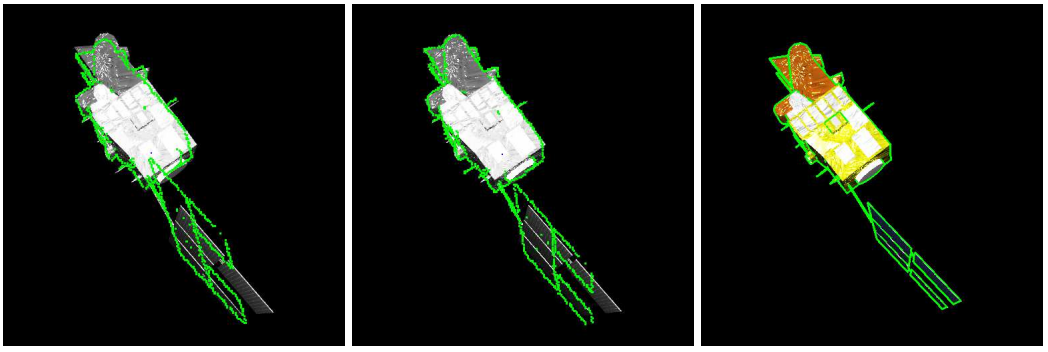


(g) Observation probabilities of the views (h) Marginal joint probabilities of the views

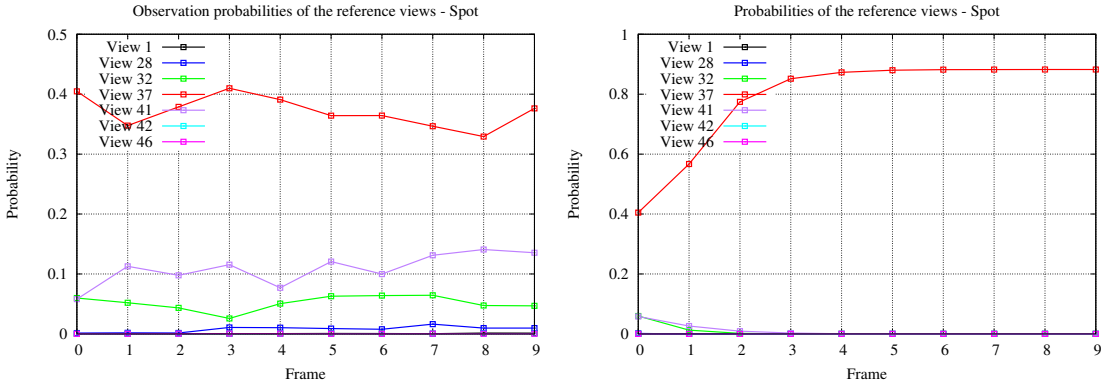
Figure 3.21 – Segmentation, detection and pose estimation process - Spot sequence 3



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 2

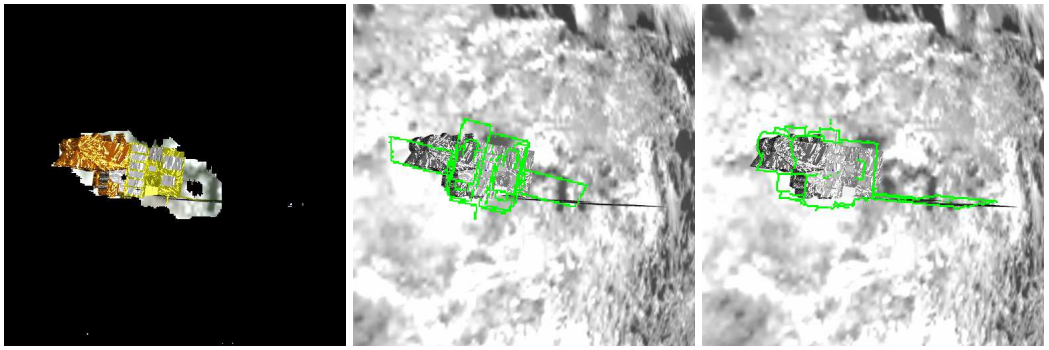


(d) Aligned view 9 (e) Pose refinement (f) Initialized tracking

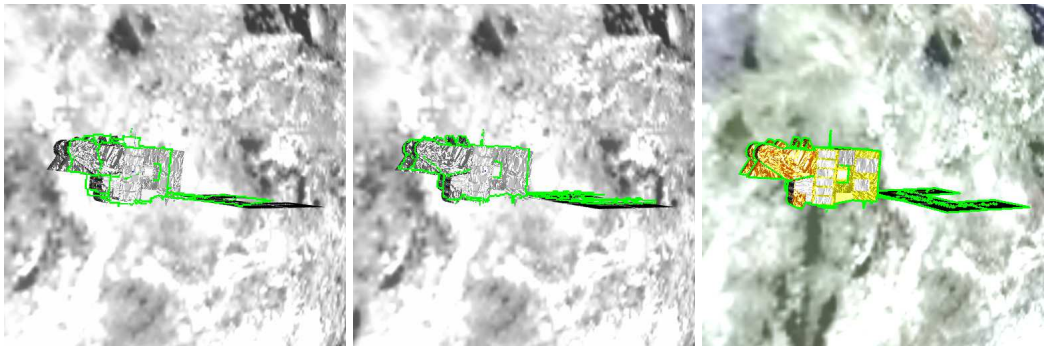


(g) Observation probabilities of the views (h) Marginal joint probabilities of the views

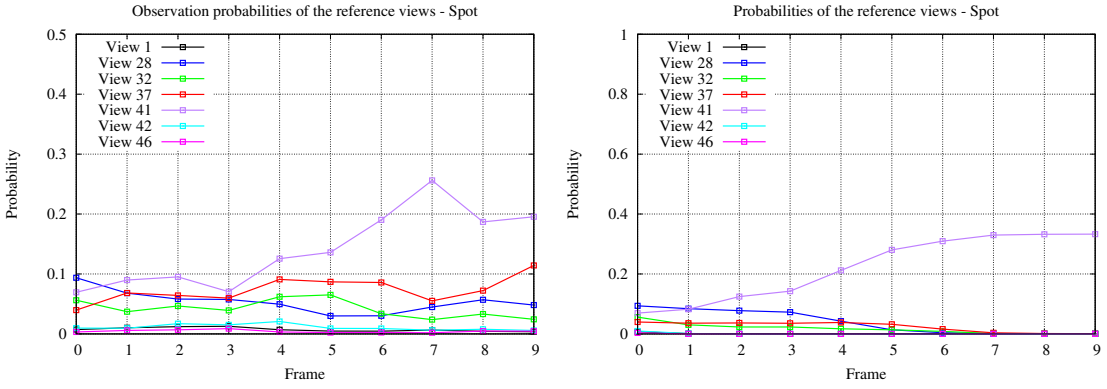
Figure 3.22 – Segmentation, detection and pose estimation process - Spot sequence 4



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 2

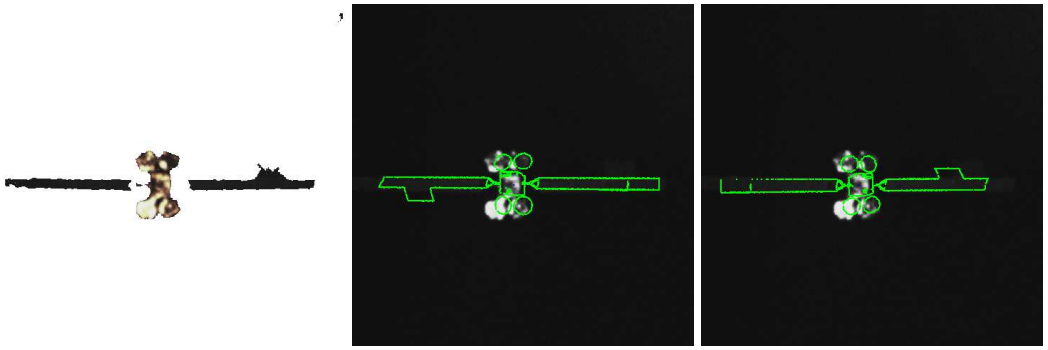


(d) Aligned view 9 (e) Pose refinement (f) Initialized tracking

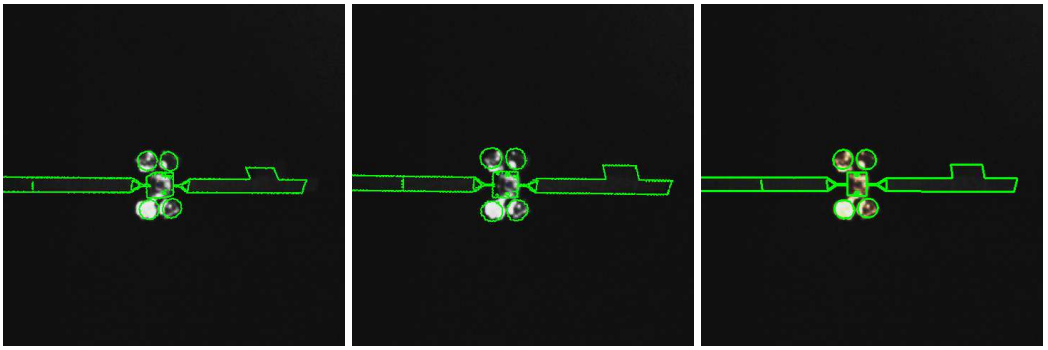


(g) Marginal joint probabilities of the views (h) Observation probabilities of the views

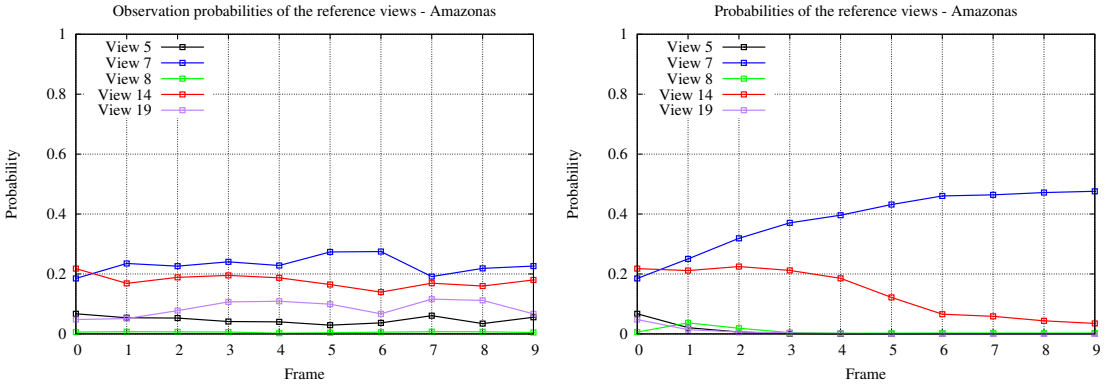
Figure 3.23 – Segmentation, detection and pose estimation process - Spot sequence 5



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 2

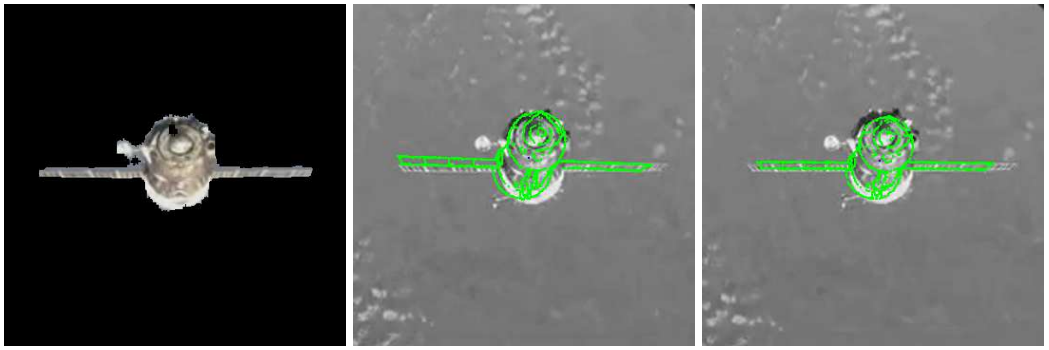


(d) Aligned view 9 (e) Pose refinement (f) Initialized tracking



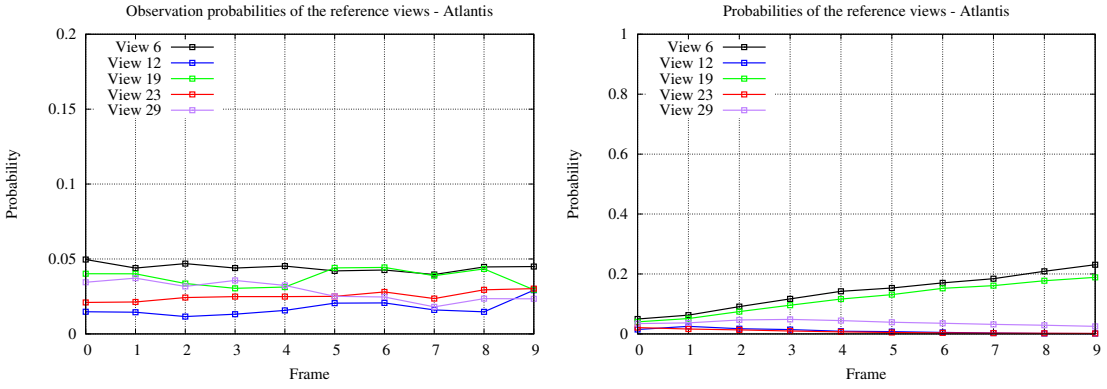
(g) Observation probabilities of the views (h) Marginal joint probabilities of the views

Figure 3.24 – Segmentation, detection and pose estimation process - Amazonas sequence



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 9

(d) Pose refinement (e) Initialized tracking

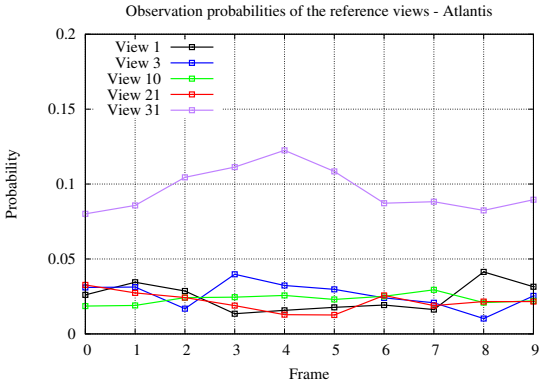


(f) Observation probabilities of the views (g) Marginal joint probabilities of the views

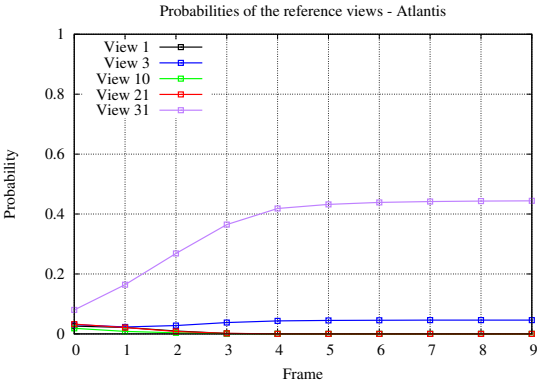
Figure 3.25 – Segmentation, detection and pose estimation process - Soyuz sequence

(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 9

(d) Pose refinement (e) Initialized tracking

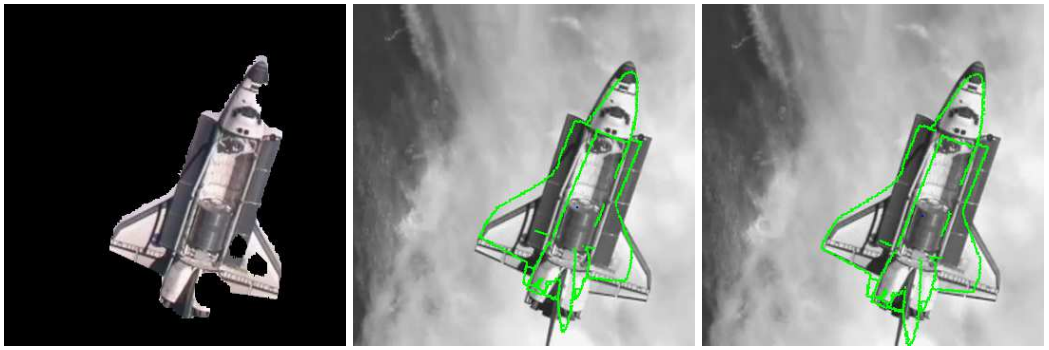


(f) Observation probabilities of the views

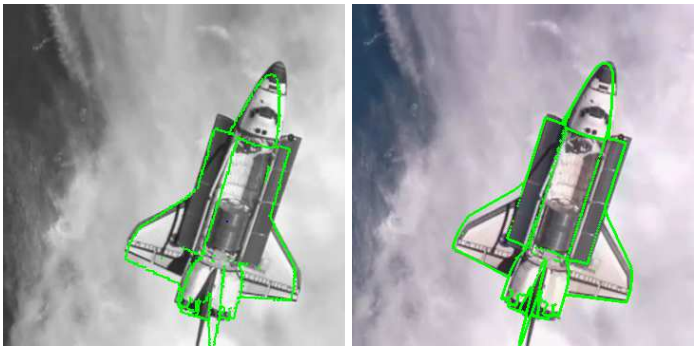


(g) Marginal joint probabilities of the views

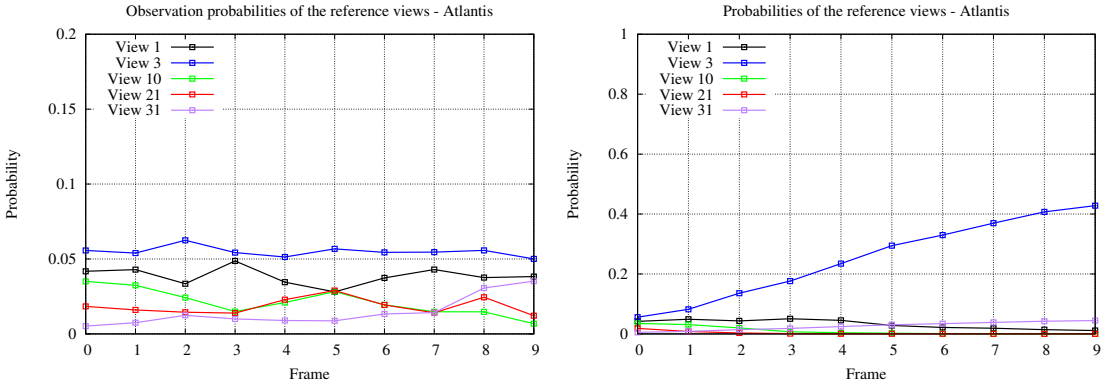
Figure 3.26 – Segmentation, detection and pose estimation process - Atlantis sequence 1



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 9



(d) Pose refinement (e) Initialized tracking

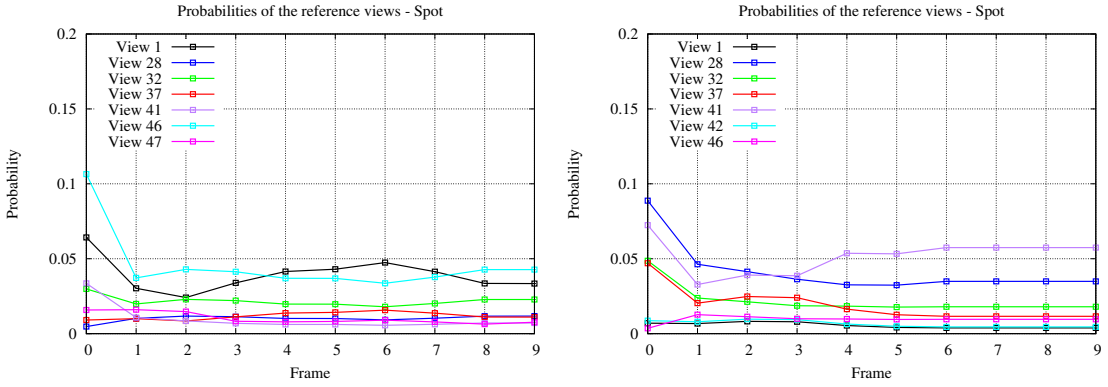


(f) Observation probabilities of the views (g) Marginal joint probabilities of the views

Figure 3.27 – Segmentation, detection and pose estimation process - Atlantis sequence 2

Influence of some tuning parameters

A focus can be paid on the influence of the parameter σ_v introduced in equation 3.3.4, which is the standard deviation accounting for the variability of viewpoints and which tunes the transition probabilities between reference views. It was set to $\sigma_v = 0.2$ rad. for the results presented above. For the Spot sequences 3 and 5 (Figures 3.21 3.23), by setting this parameter to 0.7, increasing transition probabilities, matched reference views consequently get lower marginal joint probabilities, and switches are more likely to occur, as seen on Figures 3.28(a) 3.28(b).



(a) (b)

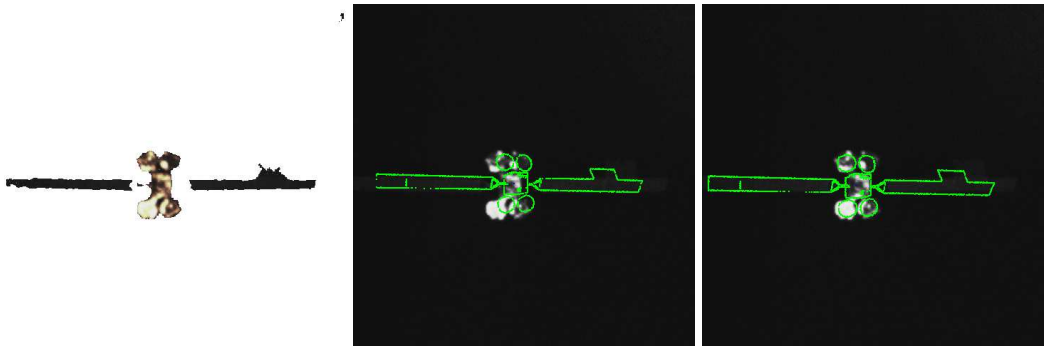
Figure 3.28 – Marginal joint probabilities of the views, for the Spot sequences 3 (a) and 5 (b) - with $\sigma_v = 0.2$ rad.

We also evaluate the effect of λ (equation(3.57) which tunes the balance between the distances between the edges and the difference between their orientations. For the sequence featuring Amazonas presented on Figure 3.24, it was set to $\lambda = 1$. For the results on Figure 3.29, it is set to $\lambda = 2$. We observe that observation probabilities are more discriminative, despite some switches between the two views 14 and 7, and the HMM rapidly accumulates evidence for view 5, showing the benefit of integrating orientations in the similarity measure.

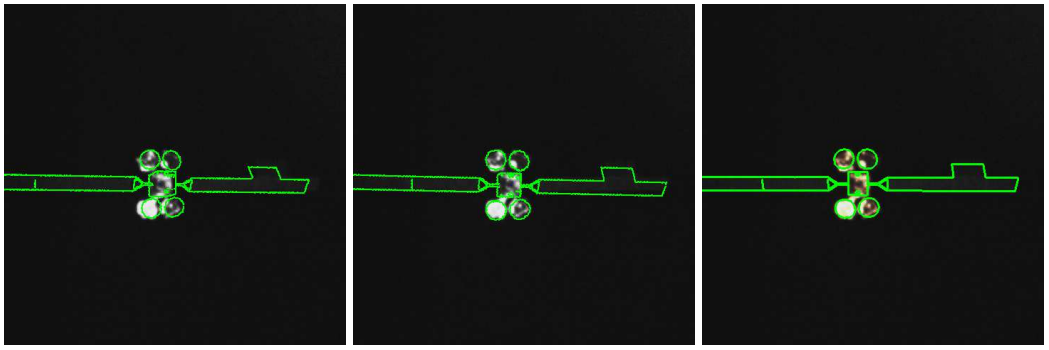
Including the segmentation phase, the overall process of matching and aligning such sets of reference views to the input image can be executed in less than $1fps$.

3.5 Conclusion

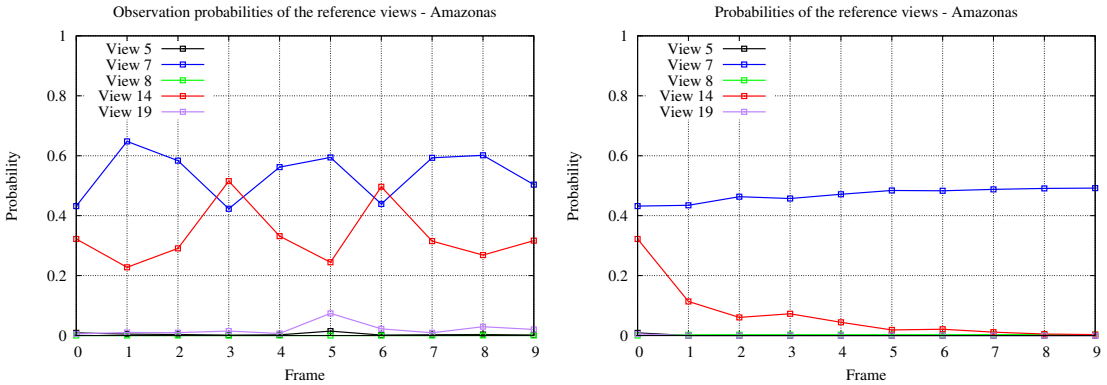
In this chapter we have described the method we propose to address the challenging issue of full-viewpoint detection and initial pose estimation in the case of complex poorly textured 3D objects such as spacecrafts. The idea is to match or align synthetic views of the 3D model with successive initial frames. In order to efficiently cover the parameter space, the views are classified into a hierarchical view graph. We also take advantage of the segmentation technique, which guides the probabilistic edge-based matching and alignment process to provide a sufficiently precise pose to initialize a classic frame-by-frame tracking. Despite the fact we have restricted this study to space objects to test and validate our



(a) Input segmented frame (b) Aligned view 0 (c) Aligned view 2



(d) Aligned view 9 (e) Pose refinement (f) Initialized tracking



(g) Observation probabilities of the views (h) Marginal joint probabilities of the views

Figure 3.29 – Segmentation, detection and pose estimation process - Amazonas sequence - $\lambda = 2$

approach, this technique could be applied to any case involving a moving camera, a single moving object and a stationary background.

In, the next chapter we present our approach to handle the second issue of the objective of this thesis, which is achieving visual localization using tracking.

Chapter 4

Pose estimation by model-based tracking

In chapter 3, we have presented our solution to handle the problem of detecting and localizing a target, based on a short sequence of initial images, captured by a camera. This process is operated at the beginning of the considered robotic task, consisting in Space rendezvous and proximity operations, between a chaser spacecraft and a target spacecraft. In this context, the target is assumed to be constantly in the field of view of the camera mounted on the chaser, throughout the approach or inspection maneuver. The localization task, initialized by the detection step, can be achieved on the next input images by using pose estimation by frame-by-frame tracking, whose principle and state-of-art have been exposed in chapter 2. As previously noted in section 1.4, the studied application involves industrial objects for which 3D CAD models can be provided. With the knowledge of the complete 3D model, and assuming that the structure of the target is consistent with this model, we suggest to address this tracking problem using a 3D model-based tracking algorithm, whose concepts and related works have been reviewed in section 2.5. Sections 4.1-4.5 present the type of model-based pose estimation framework which is adopted and the types of visual information which are processed in this work. Finally section 4.6 gives some experimental results obtained using the testing facilities presented in section 1.4.

4.1 A local non-linear optimization problem

4.1.1 Classical approaches

Our solution deals with model-based tracking, using a 3D CAD model of the target. As introduced in section 2.5.1.1, the goal is to determine an estimate $\widehat{\mathbf{r}}$ of the camera pose $\mathbf{r}, \mathbf{r} \in SE(3)$, by minimizing, with respect to $\mathbf{r}$, the forward projection error $\Delta(\mathbf{r})$. $\Delta(\mathbf{r})$ accounts for errors $e_i(\mathbf{r})$ between a set of visual features extracted from the image and the

forward projection of their 3D homologues in the image plane according to the pose:

$$\widehat{\mathbf{r}} = arg\min_{\mathbf{r}} \Delta(\mathbf{r}) \tag{4.1}$$

$$\text{with} \quad \Delta(\mathbf{r}) = \sum_i (e_i(\mathbf{r}))^2. \tag{4.2}$$

This is a non-linear minimization problem with respect to the pose parameters $\mathbf{r}$, which can be handled through a Newton-like minimization framework such as Gauss-Newton or Levenberg-Marquardt, by iteratively updating the pose $\mathbf{r}$.

Based on the knowledge of the 3D model of the target, common approaches solve this problem by using edge features [Lowe 91, Drummond 02, Vacchetti 04c, Comport 06b] as visual features to compute the set of errors $\{e_i(\mathbf{r})\}$. Edge features offer a good invariance to illumination changes or image noise. Such approaches have proven to be very efficient and various formulations of the problem have been proposed. Although one can find some differences between these solutions, the main idea is the following. Given a new image, the 3D model of the scene or the target is projected in the image according to the previous estimated camera pose $\mathbf{r}$. With classical methods, the 3D model is made of lines or segments, and each projected line $l_i(\mathbf{r}) = pr(L_i, \mathbf{r})$ of the model is then sampled, leading to a set of 2D points $\{\mathbf{x}_{i,j}\}$. Then from each sample point $\mathbf{x}_{i,j}$ a 1D search along the normal of the projected edge is performed to find a corresponding point $\mathbf{x}'_{i,j}$ in the image, as depicted on Figure 4.1.

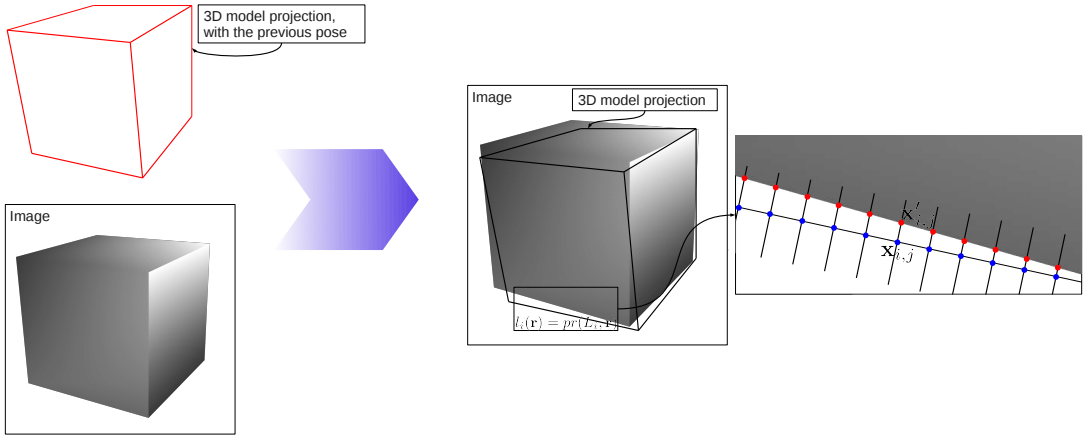


Figure 4.1 – Model projection in the image and low-level tracking through a 1D search for a corresponding edge along the normal to the model edge points.

The error function $\Delta(\mathbf{r})$ is then computed to account for distances between points $\mathbf{x}'_{i,j}$ and the projected lines l_i. It can be written as:

$$\Delta = \sum_i \sum_j (d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j}))^2 \tag{4.3}$$

where $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j})$ is the distance between a point $\mathbf{x}'_{i,j}$ in the new image and the corresponding line $l_i(\mathbf{r})$ projected in the image with a pose $\mathbf{r}$. The underlying idea of minimizing such an error function is to realign, according to the pose, the edges of the projected 3D model with edges extracted from the image.

In order to minimize the objective error function Δ with respect to the pose parameters, a Gauss-Newton approach, presented on Frame 4, is generally adopted. This is also the technique used in this work. At each iteration k, a displacement is performed in the parameter space which is $SE(3)$:

$$\mathbf{r}_{k+1} = \mathbf{r}_k \oplus \delta\mathbf{r} \tag{4.4}$$

In the manner of the Gauss-Newton minimization technique, the displacement $\delta\mathbf{r}$ is computed through:

$$\delta\mathbf{r} = -\mathbf{J}^{+}\mathbf{e} \tag{4.5}$$

where $\mathbf{J}^{+}$ is the Moore-Penrose pseudo inverse of $\mathbf{J}$, with $\mathbf{J}$ the Jacobian matrix of the error vector $\mathbf{e}(\mathbf{r})$:

$$\mathbf{J} = \frac{\partial \mathbf{e}(\mathbf{r})}{\partial \mathbf{r}} \tag{4.6}$$

$\oplus$ is an internal compositional law in the parameter space. $SE(3)$ is a Lie group (section 2.1.1), for which summation and distance cannot be applied, but there exists a one-to-one map between $SE(3)$ and associated algebra $se(3)$ defined by:

$$se(3) = \left\{ \boldsymbol{\xi} = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{v} \\ 0 & 0 \end{bmatrix} \mid [\boldsymbol{\omega}]_\times \in so(3), \boldsymbol{v} \in \mathbb{R}^3 \right\} \subset \mathbb{R}^{4\times 4}.$$

This map is the exponential map, defined in section 2.1.2.3:

$$se(3) \mapsto SE(3) \tag{4.7}$$

$$\boldsymbol{\xi} \mapsto \mathbf{M} = exp(\boldsymbol{\xi}). \tag{4.8}$$

Here, $\delta\mathbf{r} = (\boldsymbol{v}, \boldsymbol{\omega})$ denotes a screw displacement (or "velocity" of the pose), with $\boldsymbol{v}$ the translation displacement parameters and $\boldsymbol{\omega}$ the rotation displacement parameters, and the exponential map enables to express the rigid motion $\delta\mathbf{M}$ generated by $\delta\mathbf{r}$:

$$\delta\mathbf{M} = exp([\delta\mathbf{r}]) \tag{4.9}$$

$$\text{with: } [\delta\mathbf{r}] = \begin{bmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{v} \\ 0 & 0 \end{bmatrix}. \tag{4.10}$$

Based on equation (4.4), the pose $\mathbf{r}$, represented by its homogeneous matrix $\mathbf{M}_{k+1}$, can be updated as follows:

$$\mathbf{M}_{k+1} = exp([\delta\mathbf{r}])\ \mathbf{M}_k. \tag{4.11}$$

In our context, following justifications provided for the detection method (section 2.3), space objects (spacecrafts, debris) are often poorly textured objects and illumination conditions which can be encountered in space environments are variable and harsh, from dark conditions to significant specular effects due for instance to the insulating film covering the object.

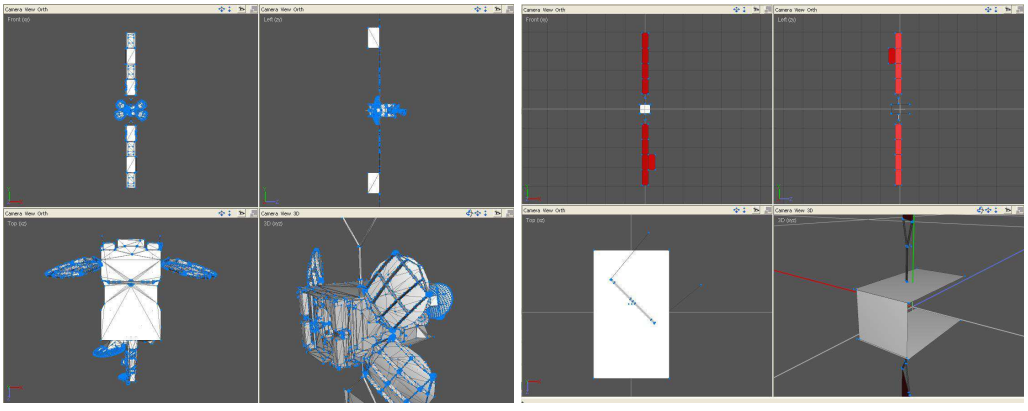
For these reasons, as a starting point, edges have been selected as the central visual information to be dealt with. Besides, the promising results obtained in [Petit 11], which is a study of the approach proposed in [Comport 06b] on a telecommunication satellite mock-up , has reinforced this choice.

However, some limitations are inherent to edges and to the classical formulation of the model-based tracking framework presented above. They are presented in section 4.1.2, and a major challenge of the thesis has been to circumvent them.

4.1.2 Limitations of classical approaches and motivations

4.1.2.1 Making model projection efficient for complex objects

A first limitation of the classical approaches regards implementation issues, since most of these techniques process polygonal 3D models which are made of segments. But achieving the model projection in the image this way faces limitations. Some problems indeed appear when dealing with objects made of cylindrical, spherical, curved or complex shapes, and space objects are likely to be made of such shapes. Furthermore, complete polygonal models for complex objects can be too heavy and need to be manually redesigned to keep the most relevant and visible edges of the scene and to make the algorithm computationally efficient. This burdensome phase had to be operated in our previous work [Petit 11]. For testing purposes, the provided 3D model of the satellite (Figure 4.2(a)) was too complex to deal with real-time applications, and its 34MB initial size had to be reduced to an acceptable size of about 10kB for real-time concerns. The goal had thus been to considerably simplify the model, by keeping the most significant geometry (Figure 4.2(b)).



(a) (b)

Figure 4.2 – Complete provided 3D model of the object (a) and example of a model used for tracking (b)

Because of this major limitation, a first challenge of the solution proposed in this thesis is to automatically process a complete polygonal 3D model. In this sense, the whole information from the geometrical shape of any kind of object or scene can be used

and a heavy phase of a manual redesign of the model is avoided. Section 4.2 presents the consequent elaborated techniques to fulfill these objectives, through the use of a 3D rendering engine and graphics acceleration.

4.1.2.2 Robustifying the pose estimation process

Though they have proven their efficiency, edges require an image segmentation process which can involve outliers and, contrary to feature points which can be specifically described, suffer from having similar appearances. It can result in ambiguities between different edges, leading to tracking failures, particularly in the case of complex objects like satellites or space debris.
Let us synthesize different techniques intended to make model-based tracking more robust, especially under different illumination, imaging conditions or scene constraints such as occlusion or background clutter. In the literature review presented in section 2.5, we have distinguished three different kinds of approaches tackling these problems of robustness.

- One solution is to combine the information provided by edges with information provided by other features, such as interest points [Masson 03, Vacchetti 04b, Rosten 05, Pressigout 07], optical flow [Brox 06, Pressigout 08], color local statistics [Panin 08b], or by additional sensors[Klein 04], see section 2.5.2.3.

- Some researches have focused on the low-level robustness. To reject outliers in the edge matching process, methods like RANSAC [Bleser 05, Choi 12] or the use of M-Estimators such as the Tukey estimator [Vacchetti 04b, Comport 06b] are common trends to make the algorithm robust to occlusions or background clutter.

  With M-estimators, [Vacchetti 04b, Comport 06b] suggest to rewrite the error function Δ as:

  $$\Delta(\mathbf{r}) = \sum_i \rho(e_i(\mathbf{r})) \tag{4.12}$$

  where ρ is a robust estimator (see section 2.5.3.2 and Frame 8). It is associated to a diagonal weight matrix $\mathbf{D}$ whose role is to specify a confidence in each feature location given the error vector $\mathbf{e}$. $\mathbf{D}$ takes the form of $\mathbf{D} = diag(w_1, ..., w_k)$ [Comport 06b]. Each w_i reflects the confidence in the i^{th} feature. Consequently, the pose update in equation (4.5) becomes:

  $$\delta\mathbf{r} = -(\mathbf{DJ})^{+}\mathbf{De}. \tag{4.13}$$

- Instead of handling a single hypothesis for a potential edge in the image, multiple hypotheses can be extracted and registered in the pose estimation [Vacchetti 04a, Teulière 10].

- Other studies have considered Bayesian filters such as Kalman filter [Yoon 08] and more recently particle filters [Klein 06, Teulière 10, Choi 12].

In this work, we have focused on these four categories to improve the state-of-the art methods.

For this purpose, some different and complementary types of robust visual features are introduced in section 4.3, a hybrid framework combining these features is proposed in section 4.4. In section 4.5 are presented tools to measure the uncertainty of the pose estimation process and some filtering and pose prediction issues are also tackled. Finally, comparative results, on both qualitative and quantitative aspects, on both synthetic and real data, are given in section 4.6.

As an overview, the general structure of the tracking and pose estimation system presented in this work can be outlined as follows:

1. Projection of the complete model with respect to the pose $\mathbf{r}_k$ computed for the previous image $\mathbf{I}_k$. To achieve this process we rely on the Graphics Process Units (GPU) and graphics libraries (openGL) that allows to perform quickly this projection regardless the complexity of the model, thanks to the use of a 3D rendering engine.

2. From the model projection we generate a set of 3D control points by back-projecting on the 3D model points corresponding to model edges or from detected interest points in the image.

3. Low-level tracking in the image to determine 3D-2D correspondences for some visual features.

4. Pose estimation step, by minimizing an objective function accounting for errors provided by the visual features.

5. Pose filtering, and pose prediction for the next frame.

4.2 Efficient projection and management of the complete CAD model

As asserted in the previous section, one of the drawbacks of classical 3D model-based methods is that all the segments making of the 3D polygonal model are treated. It implies dealing with simple objects or performing manual and heavy pre-processing on the CAD model to make it compliant with real-time or computationally efficient implementations. The approach presented in this work considers the direct use of a complete, but non necessarily polygonal model, which can be textured or untextured. We thus propose to rely on the graphics process units (GPU) and on a 3D rendering engine.

Due to their highly parallel structure, by dedicating most of their transistors to 3D graphics calculations, Graphics Process Units are designed to accelerate geometric computations on a potentially very large set of 3D vertices that build up a 3D model, to accelerate texture mapping, and to perform real-time rendering of the processed data. The significant improvements of GPU capabilities for the last 15 years and their availability on standard commercial laptops have made them a popular tool in virtual reality and more recently computer vision and augmented reality communities.

In our case, a 3D rendering engine is employed in order to automatically manage the projection of the 3D model, which can be potentially very large, and to determine the visible and prominent edges from the rendered scene. Such a method has also been considered in [Wuest 07, Reitmayr 06, Panin 08b]. An advantage of this technique is to automatically handle the hidden face removal process and to implicitly handle auto occlusions.

Since some of the considered visual features (see section 4.3) are based on edges determined from the the projection of the 3D model, an objective is to provide techniques to efficiently extract these visible and salient edges from the rendering of the complete 3D model.

Considering complex shape targets leads to forget the notion of 3D sharp edges as in [Comport 06b] or in our previous work [Petit 11] and to deal only with 3D points that belongs indifferently to sharp edges or to the "occlusion boundaries" or rims [Koenderink 90]. Two issues have then to be considered: complex model projection and 3D points selection.

As in [Wuest 07, Reitmayr 06, Panin 08b], for each new image $\mathbf{I}_{k+1}$, the model is rendered and projected using an openGL-based rendering engine (which takes advantage of the computer GPU), with respect to the previous estimated pose $\mathbf{r}_k$.

Our goal is then to obtain a set of N_g 3D points $\{\mathbf{X}_i\}_{i=1}^{N_g}$ that belong to target rims, edges and visible textures from the rendered depth buffer and textured scene. Our approach follows [Wuest 07] and is related to the techniques of silhouette generation of polygonal models described in [Isenberg 03].

4.2.1 Edge extraction from the projected model

4.2.1.1 Salient edges

Salient edges are extracted from the rendered depth or Z-buffer, which results from the projection of the 3D model. The depth buffer which corresponds to the depth values of the scene according to the camera location, at each pixel point (Figure 4.3(a)). Based on these depths we can determine the discontinuities which suit the geometrical appearance of the scene. Values given by the Z-buffer are actually non-linear with respect to the Z-coordinates in the camera frame, and are related to the near and far clipping planes, which bound the range of depth values to be rendered. Z-buffer values are normalized between 0 and 1 and the following transformation has to be applied to retrieve the true linear depth values:

$$Z(i,j) = -\frac{Z_{near} Z_{far}}{z_{buf}(i,j)(Z_{far} - Z_{near}) - Z_{far}} \tag{4.14}$$

where Z_{near} and Z_{far} are the near and far clip distances defining the clipping planes, $z_{buf}(i,j)$ is the z-buffer value and $Z(i,j)$ is the Z-coordinate in the camera frame of the projected point located at pixel (i,j). The nonlinearity means that the z-buffer values $z_{buf}(i,j)$ show better precision closer to the near clip distance, and the smaller the ratio $\frac{Z_{near}}{Z_{far}}$ is, the less precision we obtain for the depth $Z(i,j)$ close to Z_{far}. Thus these clip distances have to be precisely adapted to the object position in the camera frame. With the knowledge of a tight bounding cube of size d enclosing the 3D model and the knowledge of the Z-coordinate Z_o of the barycenter of the model, these clipping distances are set to:

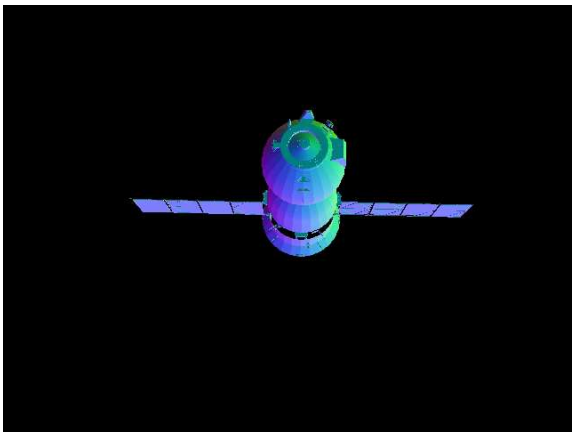
$$Z_{near} = Z_o - \frac{d}{2} \tag{4.15}$$

$$Z_{far} = Z_o + \frac{d}{2}. \tag{4.16}$$

With the aim of extracting prominent edges, we apply a second order differential operator, such as a Laplacian filter, to these computed Z values. It results in a binary edge map of the visible scene ((Figure 4.3(b)).

In our approach, we have implemented the filtering computations on the GPU through shader programming, resulting in a much lower computational time, due to the parallel structure of the process.

(a) (b)



(c)

Figure 4.3 – On (a) is represented the z-buffer of the rendered 3D model using Ogre3D, from which the edge map is computed (b).This edge map is then sampled to generate the control points. (c) shows the normal map of the rendered scene.

4.2.1.2 Texture edges

In case of highly textured scenes, geometrical edges are not sufficient and ambiguities with texture edges may occur, resulting in false matching and thus local minima during

the pose estimation process. An improvement of our method is then to combine the depth discontinuities with texture discontinuities. The rendered textures of the 3D model are thus processed by a classical Canny edge algorithm and the obtained edges are added to the ones generated from the depth buffer.

4.2.1.3 Generation of 3D control points

The steps described above provide us with the edge map of the complete scene. Dealing with the whole edge map can be computationally intensive, we propose to sample it, according to the specified number of edge control points N_g. These edge control points are thus selected on a regular grid in the image, and we finally obtain a set of N_g 2D control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$.

The 3D coordinates of the edge points in the scene or object frame can be computed by back-projecting them on the 3D model, using the Z-buffer and the pose used to project the model, based on the rigid transformation and camera projection equations presented in sections 2.2 and 2.1. With this operation, a set of N_g control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$ is generated.

Besides, the tracking phase, which will be developed in section 4.3.1.1, requires the orientations of the edges underlying the control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$. For the texture edges, the orientation is computed within the Canny algorithm on the rendered textures and it is then directly available. For the depth edges, we compute the gradients along x and y on a grayscale image of the normal map of the scene. On the normal map (see Figure 4.3(c)), the channels of the RGB values of the pixels are related to the coordinates, in the scene frame, of the normal to the corresponding surface in the scene. Since the rendering phase can suffer from aliasing, the grayscale image of the normal map is filtered using a Gaussian kernel before computing its Sobel gradients.

These basic image processing steps, as the retrieval of the normal map, are also processed on the GPU by composing vertex and fragment shaders, significantly optimizing computations.

4.3 Visual features

From the projected 3D model in the image, various visual features can then be considered to completely and pertinently represent the object. As presented in sections 2.5.2 and 4.1.2.2, two classes of visual features can be distinguished, some of them relying on geometrical distances between feature correspondences and others being based on a dense visual description using an intensity or color based similarity function. We propose to use both geometrical and intensity-based visual information, by designing three different visual features which are described hereafter. With these three visual cues, the general idea is to consistently describe the object, using its edges, its shape or silhouette and its texture through a set of interest points.

4.3.1 Geometrical edge-based features

As introduced in section 4.1 and as in many classical model-based tracking algorithms such as [Drummond 02, Vacchetti 04c, Comport 06b, Wuest 07], our solution has been first based on edges to design geometrical visual features, in order to compute a geometrical edge-based objective error function Δ^g.

4.3.1.1 Low-level tracking from the control points

The edge control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$ extracted from the projected 3D model are processed to track corresponding edges in the new image $\mathbf{I}_{k+1}$, determining a corresponding set of points $\{\mathbf{x}'_i\}_{i=1}^{N_g}$. In a similar manner to [Vacchetti 04b, Comport 06b, Wuest 07], we perform a 1D search along the normal $\mathbf{n}_i$ of the edge underlying each $\mathbf{x}_i$ (Figure 4.4). A common approach is to choose on the scan line the pixel with the maximum gradient as the matching edge point $\mathbf{x}'_i$ in the new image. The considered approach is based on the *ECM* algorithm [Bouthemy 89] for which a likelihood ratio ζ_j representing the convolution value, using a convolution mask $\mathbf{M}_\delta$, computed in $\mathbf{I}_{k+1}$ for each candidate $\mathbf{x}_{i,j}$ along the normal $\mathbf{n}_i$ and up to a range R. The maximum is selected as the corresponding point $\mathbf{x}'_i$. The convolution mask $\mathbf{M}_\delta$, classically set to a 5×5 size, is oriented according to the orientation of $\mathbf{n}_i$ in the image. A temporal constraint can also be added by evaluating the considered mask on $\mathbf{x}_i$ in the previous image $\mathbf{I}_k$ and taking into account the resulting value in ζ_j. Besides, for the selected point $\mathbf{x}'_i$, ζ_j must be greater than a certain threshold to be considered as a pertinent point and to be further taken into account in the minimization process.

4.3.1.2 Error computation and Jacobian matrix

A distance to a line

Once correspondences between the set of control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$ and the set of image edge points $\{\mathbf{x}'_i\}_{i=1}^{N_g}$ are established, our approach considers the distance between the projected 3D line $l_i(\mathbf{r})$ underlying the projected control point $\mathbf{x}_i(\mathbf{r})$ (projected from the 3D point $\mathbf{X}_i$) and the selected matching point $\mathbf{x}'_i$ in the image (see Figure 4.4). The error function $\Delta^g(\mathbf{r})$ to be minimized with respect to the pose $\mathbf{r}$ can be written as:

$$\Delta^g(\mathbf{r}) = \frac{1}{N_g}\sum_i \rho^g(e_i^g(\mathbf{r})) \tag{4.17}$$

$$= \frac{1}{N_g}\sum_i \rho^g(\sigma_g^{-1} d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i))) \tag{4.18}$$

with $e_i^g(\mathbf{r}) = \sigma_g^{-1} d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i))$. $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i))$ is the distance between the point $\mathbf{x}'_i$ and the corresponding line $l_i(\mathbf{r})$ (see Figure 4.4). ρ^g is a Tukey robust estimator and σ_g is a normalization factor accounting for the standard deviation of the error e_i^g. We use the estimate:

$$\widehat{\sigma_g} = \sqrt{\frac{1}{N_g}\sum_i \rho^g(d_\perp(l_i(\mathbf{r}_f), \mathbf{x}'_i))} \tag{4.19}$$

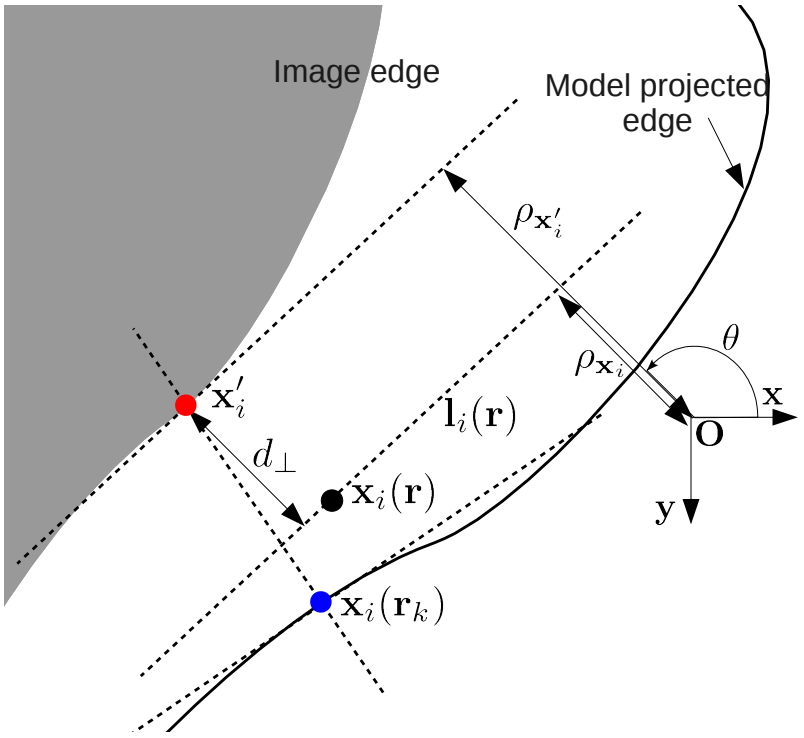


Figure 4.4 – Moving edge principle: from the initial pose $\mathbf{r}_k$, 1D search along the projected contour underlying the control point $\mathbf{x}_i$. Visual error: distance $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i) = \rho_{\mathbf{x}_i} - \rho_{\mathbf{x}'_i}$ of a point $\mathbf{x}'_i$ to a corresponding line $l_i(\mathbf{r}) = pr(L_i, \mathbf{r})$ within the minimization process. In contrast to the approach in [Comport 06b] for instance (Frame 6), control points are independently processed and a 3D line L_i is computed for each $\mathbf{x}_i$.

with $\mathbf{r}_f$ the pose computed at the end of the previous minimization process.

Remark

It has to be noted that for sharp edges the 3D points $\mathbf{X}_i(\mathbf{r})$ are not modified when we modify $\mathbf{r}$. This is no longer the case for points $\mathbf{X}_i(\mathbf{r})$ that belong to an occlusion rim or silhouette. Nevertheless, since the camera motion between two successive images is small, this approximation has no impact on the efficiency of the approach. The criterion Δ^g improves other approaches [Wuest 07, Panin 08b, Choi 12] which consider the distance between $\mathbf{x}_i(\mathbf{r})$ and $\mathbf{x}'_i$ along the 2D normal vector $\mathbf{n}_i$ to the edge underlying $\mathbf{x}_i(\mathbf{r})$, but neglecting the dependence of $\mathbf{n}_i$ w.r.t to the pose $\mathbf{r}$, what is taking into in our computation of the Jacobian matrix of e_i^g.

A key requirement is to compute the 3D equation in the scene or object frame of the line L_i such that $l_i(\mathbf{r}) = pr(L_i, \mathbf{r})$. This is necessary to perform the projection $l_i(\mathbf{r}) = pr(L_i, \mathbf{r})$ during the minimization process, to compute the error e_i^g and the corresponding Jacobian matrix $\mathbf{J}_{e_i^g}$. The computation of the equation of L_i is addressed by first expressing the polar coordinates $(\rho_{\mathbf{x}_i}, \theta_{\mathbf{x}_i})$ of $l_i(\mathbf{r}_k)$ in the image. $\rho_{\mathbf{x}_i}$ is the distance between the projected line $l_i(\mathbf{r}_k)$ and the center of the image and $\theta_{\mathbf{x}_i}$ is the angle between the image frame and the line (Figure 4.4):

$$x \cos\theta_{\mathbf{x}_i} + y \sin\theta_{\mathbf{x}_i} = \rho_{\mathbf{x}_i}, \forall (x, y) \in l_i(\mathbf{r}_k). \tag{4.20}$$

From the model rendering phase, we know $\theta_{\mathbf{x}_i}$ through the gradient computations intro-

duced in section 4.2.1 and we can compute $\rho_{\mathbf{x}_i}$, since $\mathbf{x}_i(\mathbf{r}_k) \in l_i(\mathbf{r}_k)$. Then, thanks to the normal map (see Figure 4.3(c)) the equation of the surface underlying l_i is retrieved, and it is finally quite straightforward to obtain the 3D equation of L_i in the scene frame.

l_i can thus be projected with respect to the pose $\mathbf{r}$, updating $\rho_{\mathbf{x}_i}$ and $\theta_{\mathbf{x}_i}$. The distance $d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i))$, related to the error e_i^g, can then be computed as follows (Figure 4.4):

$$d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i) = \rho_{\mathbf{x}_i} - \rho_{\mathbf{x}'_i}. \tag{4.21}$$

where

$$\rho_{\mathbf{x}'_i} = x_{\mathbf{x}'_i} \cos\theta_{\mathbf{x}_i} + y_{\mathbf{x}'_i} \sin\theta_{\mathbf{x}_i} \tag{4.22}$$

with $x_{\mathbf{x}'_i}$ and $y_{\mathbf{x}'_i}$ being the image coordinates of $\mathbf{x}'_i$.

Jacobian matrix

An issue consists in the derivation of the Jacobian matrix $\mathbf{J}_{e_i^g}$. Its definition is given by:

$$\mathbf{J}_{e_i^g} = \frac{\partial e_i^g}{\partial \mathbf{r}} = \frac{\partial d_\perp(l_i(\mathbf{r}), \mathbf{x}'_i)}{\partial \mathbf{r}}. \tag{4.23}$$

Using equation (4.21), $\mathbf{J}_{e_i^g}$ can be expressed as:

$$\begin{aligned} \mathbf{J}_{e_i^g} &= \frac{\partial \rho_{\mathbf{x}_i}}{\partial \mathbf{r}} - \frac{\partial \rho_{\mathbf{x}'_i}}{\partial \mathbf{r}} \qquad (4.24) \\ &= \frac{\partial \rho_{\mathbf{x}_i}}{\partial \mathbf{r}} + (x_{\mathbf{x}'_i} \sin\theta_{\mathbf{x}_i} - y_{\mathbf{x}'_i} \cos\theta_{\mathbf{x}_i}) \frac{\partial \theta_{\mathbf{x}_i}}{\partial \mathbf{r}} \\ &= \frac{\partial \rho_{\mathbf{x}_i}}{\partial \mathbf{r}} + \alpha \frac{\partial \theta_{\mathbf{x}_i}}{\partial \mathbf{r}} \\ &= \mathbf{J}_{\rho_{\mathbf{x}_i}} + \alpha \mathbf{J}_{\theta_{\mathbf{x}_i}} \qquad (4.25) \end{aligned}$$

with $\alpha = x_{\mathbf{x}'_i} \sin\theta_{\mathbf{x}_i} - y_{\mathbf{x}'_i} \cos\theta_{\mathbf{x}_i}$. $\mathbf{J}_{\rho_{\mathbf{x}_i}}$ and $\mathbf{J}_{\theta_{\mathbf{x}_i}}$ can be derived from [Espiau 92] where the interaction matrix related to a straight line is provided, for visual servoing purposes. It gives:

$$\begin{aligned} \mathbf{J}_{\theta_{\mathbf{x}_i}} &= \begin{bmatrix} \lambda_{\theta_{\mathbf{x}_i}} \cos\theta_{\mathbf{x}_i} & \lambda_{\theta_{\mathbf{x}_i}} \sin\theta_{\mathbf{x}_i} & -\lambda_{\theta_{\mathbf{x}_i}} \rho_{\mathbf{x}_i} & \rho_{\mathbf{x}_i} \cos\theta_{\mathbf{x}_i} & -\rho_{\mathbf{x}_i} \sin\theta_{\mathbf{x}_i} & -1 \end{bmatrix} \qquad (4.26) \\ \mathbf{J}_{\rho_{\mathbf{x}_i}} &= \begin{bmatrix} \lambda_{\rho_{\mathbf{x}_i}} \cos\theta_{\mathbf{x}_i} & \lambda_{\rho_{\mathbf{x}_i}} \sin\theta_{\mathbf{x}_i} & -\lambda_{\rho_{\mathbf{x}_i}} \rho_{\mathbf{x}_i} & (1+\rho_{\mathbf{x}_i}^2) \sin\theta_{\mathbf{x}_i} & -(1+\rho_{\mathbf{x}_i}^2) \cos\theta_{\mathbf{x}_i} & 0 \end{bmatrix} \end{aligned}$$

with

$$\begin{aligned} \lambda_{\rho_{\mathbf{x}_i}} &= \frac{(A\rho_{\mathbf{x}_i} \cos\theta_{\mathbf{x}_i} + B \sin\theta_{\mathbf{x}_i} + C)}{D} \\ \lambda_{\theta_{\mathbf{x}_i}} &= \frac{(A\rho_{\mathbf{x}_i} \sin\theta_{\mathbf{x}_i} - B \cos\theta_{\mathbf{x}_i} + C)}{D}. \end{aligned}$$

$AX + BY + CZ + D = 0$ is the equation of a 3D plane which the 3D line L_i belongs to. The normal map ((Figure 4.3(c)) of the rendered 3D model directly gives us the coefficients A, B and C, D being computed thanks to the 3D coordinates, in the scene or object frame, of the control point $\mathbf{X}_i$.

From (4.25) and (4.26) $\mathbf{J}_{e_i^g}$ takes the following form:

$$\mathbf{J}_{e_i^g} = \begin{bmatrix} \lambda_{d_\perp} \cos\theta_{\mathbf{x}_i} \\ \lambda_{d_\perp} \sin\theta_{\mathbf{x}_i} \\ -\lambda_{d_\perp}\rho_{\mathbf{x}_i} \\ (1+\rho_{\mathbf{x}_i}^2)\sin\theta_{\mathbf{x}_i} - \alpha\rho_{\mathbf{x}_i}\cos\theta_{\mathbf{x}_i} \\ -(1+\rho_{\mathbf{x}_i}^2)\cos\theta_{\mathbf{x}_i} - \alpha\rho_{\mathbf{x}_i}\sin\theta_{\mathbf{x}_i} \\ -\alpha \end{bmatrix}^T \tag{4.27}$$

with $\lambda_{d_\perp} = \lambda_{\rho_{\mathbf{x}_i}} + \alpha\lambda_{\theta_{\mathbf{x}_i}}$.

4.3.1.3 Multiple-hypotheses framework

In order to improve the robustness of the pose estimation and to avoid problems due to ambiguities between edges, it is possible to consider and register different hypotheses corresponding to potential edges. They correspond to different local extrema resulting from the *ECM* algorithm, described in section 4.3.1.2, along the scan line (see Figure 4.5). As in [Vacchetti 04b], we choose the hypothesis which has the closest distance to the projected 3D line l_i during the minimization process. The error function becomes :

$$\Delta^g = \sum_i \frac{1}{N_g} \rho^g(\sigma_g^{-1} \min_j d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j})) \tag{4.28}$$

where points $\mathbf{x}'_{i,j}$ are the selected candidates for each control point $\mathbf{x}_i$. For σ_g, this time we use the estimate: $\widehat{\sigma_g} = \sqrt{\frac{1}{N_g}\sum_i \rho^g(\min_j d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j}))}$.

Multiple-hypotheses framework with line clustering

We have also elaborated a novel multiple-hypotheses solution intended to improve robustness. This approach extends the one presented above, by taking advantage of some elements proposed in [Teulière 10]. The idea presented in the previous section was to consider and register different hypotheses corresponding to potential edges. But the projected model edge points $\{\mathbf{x}_i\}_{i=1}^{N_g}$ are treated independently, regardless their membership to primitives such as lines or particular curves. To overcome this issue, the idea is to cluster the model edge points into different primitives and to register different hypotheses consistently with these primitives. Here, we restrict ourself to line primitives, for computational reasons, and because objects considered in our applications are likely to contain lines.

Clustering model edge points into lines: from the edge map provided by the projection of the 3D model, a set of N_l 2D line segments $\{\mathbf{l}^i\}_{i=1}^{N_l}$ is locally extracted using a Hough line detector [Duda 72]. A model edge point $\mathbf{x}_k$ for which the distance to the closest line is under a certain threshold is associated to this line. We obtain a set of clusters $\{\mathbf{C}^i\}_{i=1}^{N_l}$ of model edge points corresponding to the extracted lines $\{\mathbf{l}^i\}_{i=1}^{N_l}$.

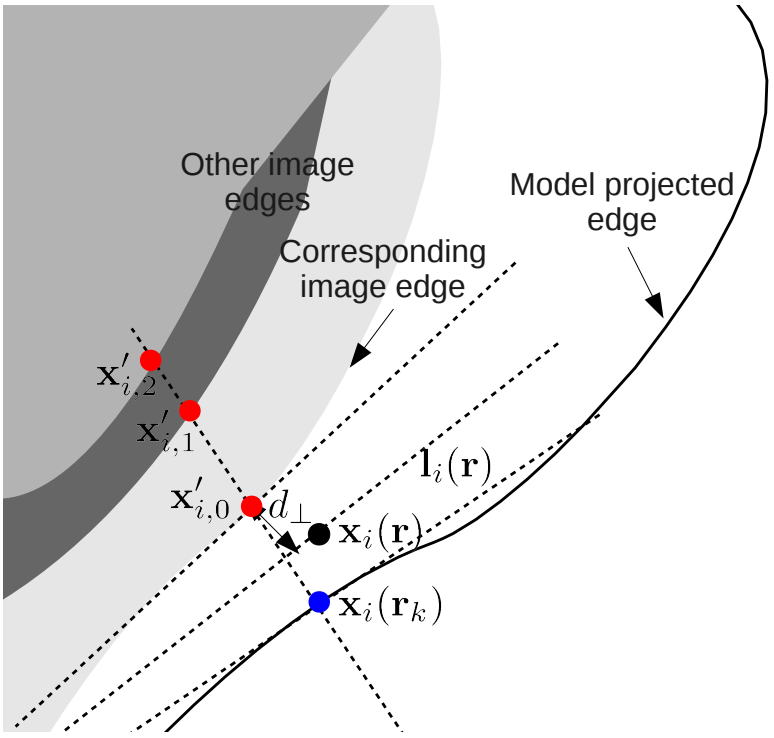


Figure 4.5 – Multiple hypotheses framework

Multiple-hypotheses registration: for each cluster $\mathbf{C}^i$, we process in a similar manner to [Teulière 10]. For a point $\mathbf{x}_{i,j}$ in $\mathbf{C}^i$, we consider several edge hypotheses $\mathbf{x}'_{i,j,l}$ (see Figure 4.6). These candidates are then classified into k_i sets of points or classes $\{c^i_m\}_{m=1}^{k_i}$ using the *k-mean* algorithm. Each c^i_m is then represented by a mean line l^i_m, which best fits the points of c^i_m, and a corresponding weight w^i_m. w^i_m represents the likelihood of class c^i_m with respect to the others in $\mathbf{C}^i$.

In [Teulière 10], random weighted draws are then performed in order to get several hypotheses on the pose. In our case, since it is time consuming, by requiring a pose estimation step for each hypothesis, we simply use here the weights w^i_m to determine the probability $\pi_{i,j,l}$ of a candidate $\mathbf{x}'_{i,j,l}$ to belong to a line. If $c^i_{m_l}$ denotes the class including $\mathbf{x}'_{i,j,l}$, we have:

$$\pi_{i,j,l} = p(c^i_{m_l} \cap \mathbf{x}'_{i,j,l}) = p(c^i_{m_l}) p(\mathbf{x}'_{i,j,l} \mid c^i_{m_l}) \tag{4.29}$$

with $p(c^i_{m_l}) \propto w^i_{m_l}$ and where the probability $p(\mathbf{x}_{i,j,l} \mid c^i_{m_l})$ is related to the distance between $\mathbf{x}'_{i,j,l}$ and the mean line $l^i_{m_l}$ associated to $c^i_{m_l}$. The function corresponding to the points $\mathbf{x}_{i,j}$ clustered into the line classes $\{\mathbf{C}^i\}_{i=1}^{N_l}$ can be written as:

$$\Delta^g_0 \propto \sum_i \sum_j \pi_{i,j,l_m} \rho^g_0(d_\perp(l_{i,j}(\mathbf{r}), \mathbf{x}'_{i,j,l_m})) \tag{4.30}$$

$$\text{with } l_m = arg\min_l (d_\perp(l_{i,j}(\mathbf{r}), \mathbf{x}'_{i,j,l})) \tag{4.31}$$

with $l_{i,j}$ the projected line, for pose $\mathbf{r}$, underlying $\mathbf{x}_{i,j}$. For the remaining points $\mathbf{x}_i$ which have not been classified into line clusters, we apply the multiple hypotheses approach

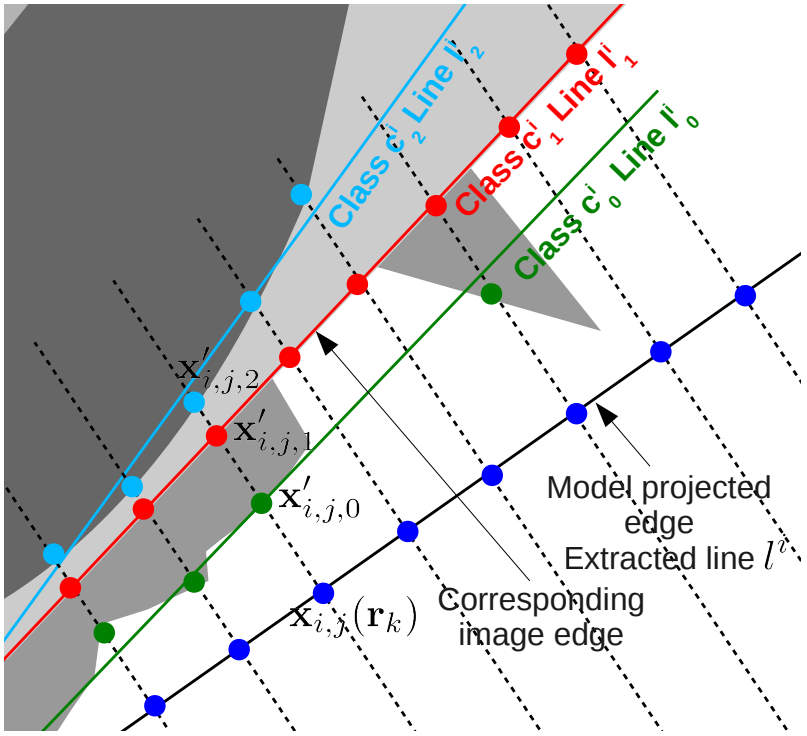


Figure 4.6 – Multiple hypotheses framework. Points $\mathbf{x}_{i,j}$ (blue dots) form the cluster $\mathbf{C}^i$ corresponding to the extracted line l^i. For a point $\mathbf{x}_{i,j}$, several hypotheses $\mathbf{x}'_{i,j,l}$ are registered, and are used to build classes c^i_m tied to lines l^i_m. The hypotheses in the class c^i_1 (red dots), which matches l^i, will have higher weights than the hypotheses of classes c^i_0 and c^i_2 (green and light blue dots), which correspond to clutter. Thus model edge points $\mathbf{x}_{i,j}$ will more likely converge towards the hypotheses of class c^i_1.

proposed in equation (4.28), giving an objective function Δ_1^g:

$$\Delta_1^g \propto \sum_i \rho_1^g(\min_j d_\perp(l_i(\mathbf{r}), \mathbf{x}'_{i,j})) \tag{4.32}$$

and Δ^g is finally simply written as:

$$\Delta^g = \Delta_0^g + \Delta_1^g. \tag{4.33}$$

4.3.2 Intensity-based features along silhouette edges

When dealing with potentially weakly textured objects, instead of defining edges in terms of gradient maxima in the image, they can be characterized by the separation between their both sides, in terms of luminance or colors of the pixels. Following this idea, a color-based error function Δ^c has been designed, based on the edges of the projected 3D model.

The technique we have used to address this problem refers to the Contracting Curve Algorithm [Hanek 04], tried out by [Panin 06, Panin 08b] for 3D tracking purposes, and which is also similar to the approaches proposed by [Brox 06, Prisacariu 12].

In order to compute Δ^c, as in [Panin 06, Panin 08b], we restrict ourself to edges belonging to the silhouette of the object. Indeed the separation between the object and the background makes more sense than for internal crease or texture edges and it limits the computational burden. The underlying idea is thus to describe how the shape or the silhouette of the object in the image can be segmented from the background. The foreground/background segmentation method presented in section 3.1 is handled with a global energy minimization framework on the whole set of pixels, with very few a priori knowledge. In contrast, the goal is here to finely perform this segmentation or separation task with respect to the pose $\mathbf{r}$ and locally, based on an a priori knowledge, which is the pose computed for the previous frame. It is particularly suitable in our context of space objects since color contrast is likely to be observed between the object and a potentially deep space black background.

More precisely, we densely consider pixel colors or simply luminance in the vicinity of the projected model silhouette edges as image features, with the goal of providing more accuracy. The principle is then to compute local color statistics (means and covariances) along the normals to the projected silhouette edges, on both sides of the edges. For each pixel along the normals, we determine a residual which represents the consistency of the pixel with these statistics, according to a fuzzy membership rule to each side. A contribution we suggest consists in adding consistency with respect to the color statistics computed on the previous frame, providing a temporal constraint.

4.3.2.1 Computation of color local statistics

More formally, a set of N_s model edge control points $\{\mathbf{x}_i\}_{i=1}^{N_s}$ belonging to the silhouette of the projected 3D model is determined from the whole set of control points $\{\mathbf{x}_i\}_{i=1}^{N_g}$,

resulting from the step described in section 4.2. From the set $\{\mathbf{x}_i(\mathbf{r})\}_{i=1}^{N_s}$, we compute color statistics up to the 2^{nd} order, on both side (object side O and background side B) of the edge underlying each $\mathbf{x}_i$. In order to handle this task, we use $2D+1$ pixels along the edge normal $\mathbf{n}_i$, regularly sampled up to a distance L (see Figure 4.7). For the object side, we obtain:

$$\nu_i^{0,O} = \sum_{j=-D}^{D} \mu_{i,j}^{O} \tag{4.34}$$

$$\boldsymbol{\nu}_i^{1,O} = \sum_{j=-D}^{D} \mu_{i,j}^{O} \mathbf{I}(\mathbf{y}_{i,j}) \tag{4.35}$$

$$\boldsymbol{\nu}_i^{2,O} = \sum_{j=-D}^{D} \mu_{i,j}^{O} \mathbf{I}(\mathbf{y}_{i,j}) \mathbf{I}(\mathbf{y}_{i,j})^T. \tag{4.36}$$

$\nu_i^{0,O}$, $\boldsymbol{\nu}_i^{1,O}$ and $\boldsymbol{\nu}_i^{2,O}$ respectively refer to the 0,1^{st} and 2^{nd} moments of the pixel intensities along the normal $\mathbf{n}_i$. $\mathbf{y}_{i,j} = \mathbf{x}_i(\mathbf{r}) + L\,\overline{d}\,\mathbf{n}_i$ are the pixels located on both sides. $\overline{d} = \frac{j}{D}$ is the normalized signed distance to $\mathbf{x}_i(\mathbf{r})$. $\mathbf{I}(\mathbf{y}_{i,j}) = \begin{bmatrix} R_i(\mathbf{y}_{i,j}) & G_i(\mathbf{y}_{i,j}) & B_i(\mathbf{y}_{i,j}) \end{bmatrix}^T$ is the RGB color vector of pixel $\mathbf{y}_{i,j}$ and $\mu_{i,j}^{O}$ are local weights giving a higher confidence on the object side, close to the edge (see [Hanek 04]). Let us note that $\nu_i^{0,O}$ is a scalar, $\boldsymbol{\nu}_i^{1,O}$ is vector of size 3 (RGB) and $\boldsymbol{\nu}_i^{2,O}$ is a 3×3 matrix. As in [Panin 08b], these statistics are then blurred with respect to the other silhouette points, for smoothness concerns. This step provide moments $\tilde{\nu}_i^{0,O}$, $\tilde{\boldsymbol{\nu}}_i^{1,O}$ and $\tilde{\boldsymbol{\nu}}_i^{2,O}$:

$$\tilde{\nu}_i^{0,O} = \sum_j e^{-\lambda|i-j|} \nu_j^{0,O} \tag{4.37}$$

$$\tilde{\boldsymbol{\nu}}_i^{1,O} = \sum_j e^{-\lambda|i-j|} \boldsymbol{\nu}_j^{1,O} \tag{4.38}$$

$$\tilde{\boldsymbol{\nu}}_i^{2,O} = \sum_j e^{-\lambda|i-j|} \boldsymbol{\nu}_j^{2,O}. \tag{4.39}$$

These moments are then normalized, to define RGB means $\overline{\mathbf{I}}_i^O$ and covariances $\overline{\mathbf{R}}_i^O$ for each $\mathbf{x}_i(\mathbf{r})$:

$$\overline{\mathbf{I}}_i^O = \frac{\tilde{\boldsymbol{\nu}}_i^{1,O}}{\tilde{\nu}_i^{0,O}} \quad \text{and} \quad \overline{\mathbf{R}}_i^O = \frac{\tilde{\boldsymbol{\nu}}_i^{2,O}}{\tilde{\nu}_i^{0,O}}. \tag{4.40}$$

We proceed the same way for the background B, leading to:

$$\tilde{\boldsymbol{\nu}}_i^{k,B} = \sum_j e^{-\lambda|i-j|} \boldsymbol{\nu}_j^{k,B}, k = 0,1,2 \tag{4.41}$$

$$\overline{\mathbf{I}}_i^B = \frac{\tilde{\boldsymbol{\nu}}_i^{1,B}}{\tilde{\nu}_i^{0,B}} \quad \text{and} \quad \overline{\mathbf{R}}_i^B = \frac{\tilde{\boldsymbol{\nu}}_i^{2,B}}{\tilde{\nu}_i^{0,B}}. \tag{4.42}$$

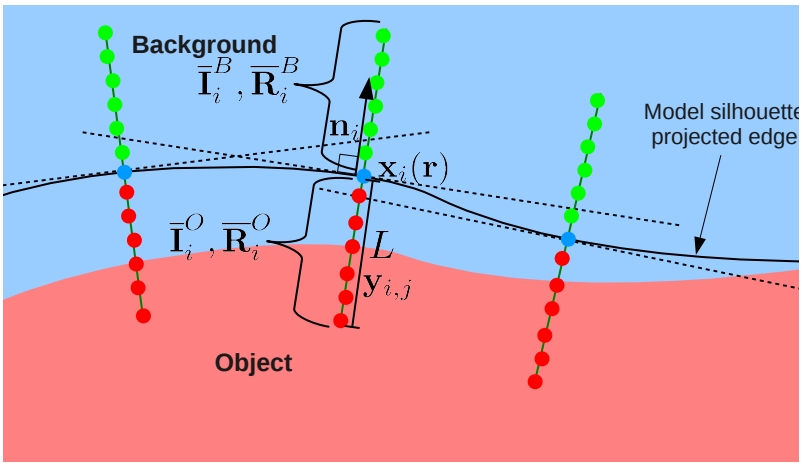


Figure 4.7 – Computation of local color statistics on both the background (B) and object (O) sides.

4.3.2.2 Error computation and Jacobian matrix

The computation of the local errors $e^c_{i,j}(\mathbf{r})$ of the error function $\Delta^c(\mathbf{r})$ is based on on the consistency of observed color components of pixels $\mathbf{y}_{i,j}$ according to the computed color statistics. This consistency is evaluated using a function $a(\overline{d})$ as a fuzzy membership rule to the object side:

$$a(\overline{d}) = \frac{1}{2}(erf(\frac{\overline{d}}{\sqrt{2}\sigma_a} + 1), \overline{d} = -1..1 \tag{4.43}$$

erf being the Gauss error function [Abramowitz 64] whose shape can be seen on Figure 4.8. σ_a is a standard deviation defining the sharpness of the membership rule. Both object and background statistics can thus be mixed, resulting in means $\hat{\mathbf{I}}_{i,j}(\mathbf{r})$ and covariances $\hat{\mathbf{R}}_{i,j}(\mathbf{r})$, smoothed along the normal $\mathbf{n}_i$:

$$\hat{\mathbf{I}}_{i,j}(\mathbf{r}) = a(\overline{d}(\mathbf{r}))\overline{\mathbf{I}}_i^O + (1 - a(\overline{d}(\mathbf{r})))\overline{\mathbf{I}}_i^B \tag{4.44}$$

$$\hat{\mathbf{R}}_{i,j}(\mathbf{r}) = a(\overline{d}(\mathbf{r}))\overline{\mathbf{R}}_i^O + (1 - a(\overline{d}(\mathbf{r})))\overline{\mathbf{R}}_i^B \tag{4.45}$$

$\hat{\mathbf{I}}_{i,j}(\mathbf{r})$ can thus be seen as a desired color value for the j^{th} pixel $\mathbf{y}_{i,j}$ on the normal $\mathbf{n}_i$, wether it is on the object O or background side B, with $j = D\overline{d}(\mathbf{r})$, and $\hat{\mathbf{R}}_{i,j}(\mathbf{r})$ the associated covariance. $\hat{\mathbf{I}}_{i,j}(\mathbf{r})$ is vector of size 3 (RGB)and $\hat{\mathbf{R}}_{i,j}(\mathbf{r})$ is a 3×3 matrix.

The error $e^c_{i,j}(\mathbf{r})$ can then be defined as:

$$e^c_{i,j}(\mathbf{r}) = (\hat{\mathbf{I}}_{i,j}(\mathbf{r}) - \mathbf{I}(\mathbf{y}_{i,j}))^T\hat{\mathbf{R}}^{-1}_{i,j}(\hat{\mathbf{I}}_{i,j}(\mathbf{r}) - \mathbf{I}(\mathbf{y}_{i,j}))) \tag{4.46}$$

$$= \mathbf{f}^c_{i,j}(\mathbf{r})^T\hat{\mathbf{R}}_i^{-1}\mathbf{f}^c_{i,j}(\mathbf{r}) \tag{4.47}$$

with $\mathbf{f}^c_{i,j}(\mathbf{r}) = \hat{\mathbf{I}}_i(\mathbf{r}) - \mathbf{I}(\mathbf{y}_{i,j})$. With the error $e^c_{i,j}(\mathbf{r})$, the general idea is to optimize the position $\overline{d}(\mathbf{r})$ of the membership rule a along the normal, so that the desired value $\hat{\mathbf{I}}_i(\mathbf{r})$ best matches the actual value $\mathbf{I}(\mathbf{y}_{i,j})$, minimizing $\mathbf{e}^c_{i,j}(\mathbf{r})$. Let us note that the dependence of $\hat{\mathbf{R}}_{i,j}(\mathbf{r})$ w.r.t. the pose $\mathbf{r}$ is neglected to reduce computational costs.
In order to cope with possible outliers and to improve robustness with respect to occlusion noise or clutter, we propose to integrate a Tukey M-estimator in Δ^c, which becomes:

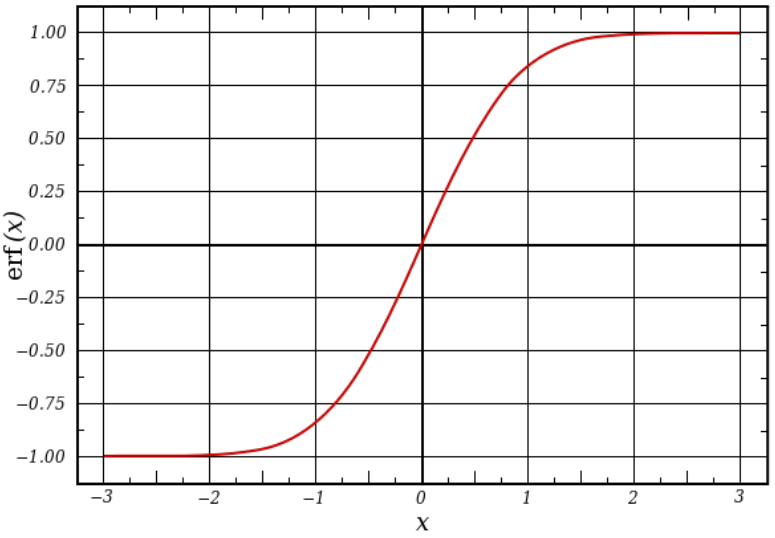


Figure 4.8 – Profile of the error function erf.

$$\Delta^c(\mathbf{r}) = \frac{1}{N_c}\sum_i \rho^c\Big(\sqrt{\sum_j e^c_{i,j}(\mathbf{r})}\Big) \tag{4.48}$$

with $N_c = (2D+1)N_s$ accounting for the number of color features. ρ^c is applied to a $N_s \times 1$ vector accounting for errors computed for each silhouette edge control point $\mathbf{x}_i(\mathbf{r})$, reflecting the quality of the separation, in terms of color likelihood, between the object and the background at this particular point, by summing the errors along $\mathbf{n}_i$.

The Jacobian matrix $\mathbf{J}_{e^c_{i,j}}$ can be computed as follows:

$$\mathbf{J}_{e^c_{i,j}} = \frac{\partial e^c_{i,j}(\mathbf{r})}{\partial \mathbf{r}} \tag{4.49}$$

$$= \frac{1}{2e^c_{i,j}}\frac{\partial \mathbf{f}^c_{i,j}(\mathbf{r})}{\partial \mathbf{r}}(\hat{\mathbf{R}}_i^{-1} + \hat{\mathbf{R}}_i^{-T})\mathbf{f}^c_{i,j}(\mathbf{r}) \tag{4.50}$$

Since $\hat{\mathbf{R}}_i^{-1}$ is a symmetric matrix, we have:

$$\mathbf{J}_{e^c_{i,j}} = \frac{1}{e^c_{i,j}}\Big(\frac{\partial \mathbf{f}^c_{i,j}(\mathbf{r})}{\partial \mathbf{r}}\Big)^T \hat{\mathbf{R}}_i^{-1}\mathbf{f}^c_{i,j}(\mathbf{r}) \tag{4.51}$$

The Jacobian matrix $\mathbf{J}_{\mathbf{f}^c_{i,j}} = \frac{\partial \mathbf{f}^c_{i,j}(\mathbf{r})}{\partial \mathbf{r}}$ is computed as:

$$\mathbf{J}_{\mathbf{f}^c_{i,j}} = \frac{\partial \hat{\mathbf{I}}_{i,j}(\mathbf{r})}{\partial \mathbf{r}} \tag{4.52}$$

$\bar{\mathbf{I}}_i^O$ and $\bar{\mathbf{I}}_i^B$ are updated at each iteration of the minimization process. However, to reduce the complexity, their dependence on the pose $\mathbf{r}$ are neglected.

$$= (\bar{\mathbf{I}}_i^O - \bar{\mathbf{I}}_i^B)\frac{\partial a(\bar{d}(\mathbf{r}))}{\partial d}\frac{\partial d}{\partial \mathbf{r}} \tag{4.53}$$

As in [Panin 08b, Choi 12], we have:

$$\frac{\partial d}{\partial \mathbf{r}} = \frac{1}{L}\mathbf{n}_i^T \mathbf{J}_{\mathbf{x}_i}. \tag{4.54}$$

The dependence of $\mathbf{n}_i$ on $\mathbf{r}$ is also neglected, saving computations. $\mathbf{J}_{\mathbf{x}_i} = \frac{\partial \mathbf{x}_i(\mathbf{r})}{\partial \mathbf{r}}$ is the Jacobian matrix for a point, which is given by;

$$\mathbf{J}_{\mathbf{x}_i} = \mathbf{K}_0 \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & (1+y^2) & xy & -x \end{bmatrix} \tag{4.55}$$

with

$$\mathbf{K}_0 = \begin{bmatrix} p_x & 0 \\ 0 & p_y \end{bmatrix} \tag{4.56}$$

the focal ratio parameters of the camera. (x, y) denotes the meter coordinates of the image point $\mathbf{x}_i$, and Z the depth of the corresponding 3D point.

As stated before and as previously suggested by [Panin 06], the $2D+1$ pixels $\mathbf{y}_{i,j}$ used to compute the local color statistics are regularly sampled along the normal $\mathbf{n}_i$ with a sample step $\delta_D = \frac{L}{D}$, with L the fixed range along $\mathbf{n}_i$. Consequently $2D+1$ errors $e^c_{i,j}$ and their corresponding Jacobians $\mathbf{J}_{e^c_{i,j}}$ are computed. Increasing L would provide more robustness to large motions and blur or noise, while increasing δ_D would aim at saving computations, while being less accurate. Since the computation of local statistics is much less costly than the computations of errors and Jacobians, an alternative would be to compute these statistics with $\delta_D = 1$, while we would compute and retain errors and Jacobians with $\delta_D > 1$, improving the ratio accuracy/computations.

4.3.2.3 Temporal constraint

For more smoothness and accuracy, we introduce a temporal constraint to the objective function by considering the information of past frames. The idea is to integrate the color statistics computed on the previous frame ${}^P\mathbf{I}$ for the silhouette edge points $\mathbf{x}_i(\mathbf{r}_k)$ at the first iteration of the minimization process. This is handled by rewriting the computation of local statistics detailed in equation (4.36) the following way, for the object side:

$$\nu_i^{0,O} = \sum_{j=-D}^{D} \mu_{i,j}^O \tag{4.57}$$

$$\boldsymbol{\nu}_i^{1,O} = \sum_{j=-D}^{D} \mu_{i,j}^O (\alpha \mathbf{I}(\mathbf{y}_{i,j}) + (1-\alpha)\ {}^P\mathbf{I}(\mathbf{y}_{i,j})) \tag{4.58}$$

$$\boldsymbol{\nu}_i^{2,O} = \sum_{j=-D}^{D} \mu_{i,j}^O (\alpha \mathbf{I}(\mathbf{y}_{i,j}) + (1-\alpha)\ {}^P\mathbf{I}(\mathbf{y}_{i,j})(\alpha \mathbf{I}(\mathbf{y}_{i,j} + (1-\alpha)\ {}^P\mathbf{I}(\mathbf{y}_{i,j}))^T \tag{4.59}$$

with α a weighting factor, $0 < \alpha < 1$. The same operation si performed for the background side. From these new statistics, mixed means $\hat{\mathbf{I}}_{i,j}(\mathbf{r})$ and covariances $\hat{\mathbf{I}}_{i,j}(\mathbf{r})$, as well as errors $e^c_{i,j}$ and Jacobian matrices $\mathbf{J}_{e^c_{i,j}}$ can be consequently derived, following the

steps previously described.

Let us note that instead of using colors, these visual features and errors could also be simply based on luminance. In that case vector $\mathbf{f}_{i,j}$ and covariances $\mathbf{R}_{i,j}$ become scalars.

4.3.3 Keypoint-based features

Another class of visual features which can be used are interest points (or keypoints) tracked across the image sequence. As previously suggested by [Vacchetti 03, Brox 06] or by [Vacchetti 04b, Pressigout 08] within their hybrid approaches, the idea is to design a texture-based objective function Δ_p accounting for geometrical distances between interest points extracted and tracked or over successive images. But in contrast to [Pressigout 08], which process 2D-2D point correspondences to estimate the 2D transformation from $\mathbf{I}_k$ to $\mathbf{I}_{k+1}$ of planar local regions underlying the points. We use 3D-2D point correspondences to directly minimize Δ_p with respect to the pose $\mathbf{r}$.

More specifically, let us denote $\{\mathbf{x}_i\}_{i=1}^{N_p}$ a set of detected interests points in frame $\mathbf{I}_k$. Assuming the pose $\mathbf{r}_k$ has been properly estimated, we can restrict these points to be lying on the projected 3D model with respect to $\mathbf{r}_k$. Since we rely on a complete 3D model, the depth of the points in the scene can be accurately retrieved, and using $\mathbf{r}$, we can back-project these points on the 3D model, giving a set $\{\mathbf{X}_i\}_{i=1}^{N_p}$ of 3D points of the 3D model. This is a major difference with respect to [Vacchetti 04b] which aims at simultaneously optimizing the camera poses and projections of the matched points in two successive frames, relaxing the assumption of having an accurate previous pose estimate and an accurate 3D model, but increasing computations. Our knowledge of a complete 3D model, along with the use of convenient rendering techniques allows us the keep this assumption valid.

Approaches presented in [Brox 06, Pressigout 08] process regularly spread points lying on the 3D model [Brox 06] or on planar surfaces of the model [Pressigout 08] and determine the 2D-3D [Brox 06] or 2D-2D [Pressigout 08] correspondences by computing their optical flow from $\mathbf{I}_k$ to $\mathbf{I}_{k+1}$. These methods, though providing dense accurate information, can however be computationally challenging. Instead, we employ the Harris corners detector inside the silhouette of the projected model in the image to extract $\{\mathbf{x}_i\}_{i=1}^{N_p}$ in $\mathbf{I}_k$. Then, the KLT tracking algorithm enables to track this set of points in frame $\mathbf{I}_{k+1}$, resulting in a corresponding set $\{\mathbf{x}'_i\}_{i=1}^{N_p}$.

Error computation and Jacobian matrix

From the correspondences between $\{\mathbf{X}_i\}_{i=1}^{N_p}$ and $\{\mathbf{x}'_i\}_{i=1}^{N_p}$, Δ^p can be computed as follows:

$$\Delta^p(\mathbf{r}) = \frac{1}{N_p}\sum_{i}^{N_p}\rho^p(e_i^p) \tag{4.60}$$

with

$$e_i^p = {\sigma_p}^{-1}(\mathbf{x}_i(\mathbf{r}) - \mathbf{x}'_i) \tag{4.61}$$

$$\text{and } \mathbf{x}_i(\mathbf{r}) = pr(\mathbf{X}_i, \mathbf{r})). \tag{4.62}$$

ρ^p is the Tukey robust estimator associated to these errors. σ_p accounts for the standard deviation of errors e_i^p. Similarly to σ_g, we use the estimate:

$$\widehat{\sigma_p} = \sqrt{\frac{1}{N_p} \sum_i \rho^p(\mathbf{x}_i(\mathbf{r}_f) - \mathbf{x}_i')}. \tag{4.63}$$

The Jacobian matrix $\mathbf{J}s_i^p$ is the Jacobian matrix of a point, given by equation(4.55):

$$\mathbf{J}e_i^p = \frac{\partial e_i^p}{\partial \mathbf{r}} = \mathbf{K} \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & (1+y^2) & xy & -x \end{bmatrix} \tag{4.64}$$

with (x, y) the meter coordinates of the image point $\mathbf{x}_i = pr(\mathbf{X}_i, \mathbf{r}))$, and Z the depth of the corresponding 3D point.

4.4 Hybrid approach

Several attempts [Masson 03, Vacchetti 04b, Pressigout 07, Pressigout 08, Panin 08b] have been made to fuse different visual features to improve the tracking accuracy and robustness. In the same spirit and following ideas suggested in [Panin 08b], which fuses geometrical edge feature with color ones, and in [Vacchetti 04b], which proposes a hybrid method incorporating edge and interest point features, we suggest to integrate the different features presented in section 4.3 in the pose estimation process. The goal is to benefit from their complementarity and to overcome the limitations of classical edge-based approaches. Let us recall the specificities of these primitives:

- Geometrical features based on edges or interest points such as Harris corners rely on line-to-point or point-to-point correspondences and on geometrical distances accounting for these correspondences. Both features correspond to complementary local regions in the images. On one hand, edges have the advantage of being robust to illumination conditions but suffer from having similar appearances, resulting in ambiguities between different edges and potential local minima. On the other hand, points features can be described more specifically, with a more discriminative matching process, imposing a better spatio-temporal constraint. Though begin locally performed, the KLT algorithm allows a larger convergence radius. However, these keypoints are more sensitive to illumination conditions, for both extraction and tracking steps. Besides, since point features are based on the extraction of Harris corners which are back-projected on the 3D model, pose errors resulting from the tracking phase at the previous frame are integrated in the back-projection process, leading to potential *drift* problems across the sequence. Whereas for edges, the edge matching process in the image instead rely on the absolute reference of the projection of the 3D model.

- With color or intensity edge-based feature, the idea is to avoid any image extraction or segmentation that could lead to outliers and mismatches, especially in the case of noise, background clutter or image blur. By processing a dense information along the silhouette of the projected 3D model by modeling the color (or luminance) appearance on both sides of the edges, using simple statistics, and optimizing their

separation, a much better accuracy can be achieved. A main advantage is also a better robustness to image or motion blur, background clutter or noise. However, among drawbacks these features need color contrast to perform efficiently and are limited by their computational costs, making them restricted to silhouette contours.

By combining these different complementary cues in the pose estimation framework means integrating their respective error function into a global one. Δ can indeed be rewritten as:

$$\Delta = w^g \Delta^g + w^c \Delta^c + w^p \Delta^p \tag{4.65}$$

As defined in section 4.3, Δ^g refers to the geometrical edge-based cost function, Δ^c stands for the color-based one and Δ^p corresponds to the interest point features. w^g, w^c and w^p are the respective weighting parameters ($0 < w < 1$).

The combination of the three types of features and their respective errors $e_i^g(\mathbf{r})$, $e_{i,j}^c(\mathbf{r})$ and $e_i^p(\mathbf{r})$ in the minimization framework is achieved by stacking the error vectors $\mathbf{e}_i^g$, $\mathbf{e}_{i,j}^c$ and $\mathbf{e}_i^p$ into a global error vector $\mathbf{e}$ and their corresponding Jacobian matrices $\mathbf{J}_{\mathbf{e}^g}$, $\mathbf{J}_{\mathbf{e}^c}$ and $\mathbf{J}_{\mathbf{e}^p}$ into a global Jacobian matrix $\mathbf{J}$. By setting parameters $\lambda^g = \frac{w^g}{N_g}$, $\lambda^c = \frac{w^c}{N_c}$ and $\lambda^p = \frac{w^p}{N_p}$.

$$\mathbf{e} = \begin{bmatrix} \sqrt{\lambda^g}\mathbf{e}^g \\ \sqrt{\lambda^c}\mathbf{e}^c \\ \sqrt{\lambda^p}\mathbf{e}^p \end{bmatrix} \tag{4.66}$$

$$\text{with: } \mathbf{e}^g = \begin{bmatrix} e_1^g \\ \vdots \\ e_{N_g}^g \end{bmatrix}, \quad \mathbf{e}^c = \begin{bmatrix} e_{1,1}^c \\ \vdots \\ e_{N_s,2D}^c \end{bmatrix}, \quad \mathbf{e}^p = \begin{bmatrix} e_1^p \\ \vdots \\ e_{N_p}^p \end{bmatrix} \tag{4.67}$$

and their Jacobian matrices are given by:

$$\mathbf{J} = \begin{bmatrix} \sqrt{\lambda^g}\mathbf{J}_{\mathbf{e}^g} \\ \sqrt{\lambda^c}\mathbf{J}_{\mathbf{e}^c} \\ \sqrt{\lambda^p}\mathbf{J}_{\mathbf{e}^p} \end{bmatrix} \tag{4.68}$$

$$\text{with} \quad \mathbf{J}_{\mathbf{e}^g} = \begin{bmatrix} \mathbf{J}_{e_1^g} \\ \vdots \\ \mathbf{J}_{e_{N_g}^g} \end{bmatrix} \quad \mathbf{J}_{\mathbf{e}^c} = \begin{bmatrix} \mathbf{J}_{e_{1,1}^c} \\ \vdots \\ \mathbf{J}_{e_{N_s,2D}^c} \end{bmatrix} \quad \mathbf{J}_{\mathbf{e}^g} = \begin{bmatrix} \mathbf{J}_{e_1^p} \\ \vdots \\ \mathbf{J}_{e_{N_p}^p} \end{bmatrix} \tag{4.69}$$

$\mathbf{e}$ is a $N_g + N_c + N_p$ vector and $\mathbf{J}$ is a $(N_g + N_c + N_p) \times 6$ matrix. Regarding the weighting matrix $\mathbf{D}$, it is written as $\mathbf{D} = blockdiag(\mathbf{D^g}, \mathbf{D^c}, \mathbf{D}^p)$, where $\mathbf{D}^g$, $\mathbf{D^c}$ and $\mathbf{D^p}$ are the respective weighting matrices associated to the robust estimators ρ^g, ρ^c and ρ^p.

The computation (4.5) of the displacement for an iteration of the Gauss-Newton minimization becomes:

$$\begin{aligned} \delta\mathbf{r} &= -(\mathbf{D}\mathbf{J}_\mathbf{s})^+ \mathbf{D}\mathbf{e}(\mathbf{r}) && (4.70) \\ &= -(\lambda^g \mathbf{J}^{gT}\mathbf{D}^{gT}\mathbf{D}^g\mathbf{J}^g + \lambda^c \mathbf{J}^{cT}\mathbf{D}^{cT}\mathbf{D}^c\mathbf{J}^c + \\ &\quad \lambda^p \mathbf{J}^{pT}\mathbf{D}^{pT}\mathbf{D}^p\mathbf{J}^p)^{-1}(\lambda^g \mathbf{J}^{gT}\mathbf{D}^{gT}\mathbf{D}^g\mathbf{e}^g(\mathbf{r}) + \\ &\quad \lambda^c \mathbf{J}^{cT}\mathbf{D}^{cT}\mathbf{D}^c\mathbf{e}^c(\mathbf{r}) + \lambda^p \mathbf{J}^{pT}\mathbf{D}^{pT}\mathbf{D}^p\mathbf{e}^p(\mathbf{r})) && (4.71) \end{aligned}$$

4.5 Filtering and pose prediction

The previous section presented the general framework for our tracking system, which is for the moment based on a purely deterministic approach. However, based on the assumption that the $\mathbf{r}$ can be considered as a random variable, statistical or probabilistic tools can be integrated to improve this deterministic framework:

- The camera pose and the camera displacement between successive frames can be assumed to be random variables, following Gaussian distributions. Through covariances, their uncertainty can be characterized, propagated from the low-level uncertainty (section 4.5.1), giving an indicator of reliability of the tracking process.
- With a purely deterministic framework, pose estimation can suffer from jittering, which can potentially lead to tracking failures. In order to avoid this, a Kalman filter is proposed to smooth pose estimates (section 4.5.2.1), with the knowledge of a particular dynamic model of the system.
- Based on this dynamic model, the filtering process can be used to provide a prediction of the pose, so that the deterministic minimization can be initialized more finely.

4.5.1 Pose uncertainty as a measure of tracking integrity

An important tool to set up is the measurement of the quality and reliability of the tracking process, based on the errors provided by the different cues integrated in the objective function. For this purpose, we can compute the covariance matrix $\mathbf{\Sigma}_{\delta\mathbf{r}}$ on the parameters of the pose error $\delta\mathbf{r}$, which results from the errors $\mathbf{e}$, based on equation (4.11). We can indeed assume that the pose error $\delta\mathbf{r}$ follows a Gaussian distribution $\delta\mathbf{r} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{\delta\mathbf{r}})$. We also assume that error $\mathbf{e}$ follows a Gaussian distribution $\mathbf{e} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{\mathbf{e}})$, with $\mathbf{\Sigma}_{\mathbf{e}} = blockdiag(\lambda^g \mathbf{I}_{N_g \times N_g}, \lambda^c \mathbf{I}_{N_c \times N_c}, \lambda^p \mathbf{I}_{N_p \times N_p})$, since the errors are normalized. From equation (4.5) giving the value of $\delta\mathbf{r}$, $\mathbf{\Sigma}_{\delta\mathbf{r}}$ can be written as:

$$\begin{aligned} \mathbf{\Sigma}_{\delta\mathbf{r}} &= E\left[\delta\mathbf{r}\delta\mathbf{r}^T\right] \\ &= E\left[(\mathbf{DJ})^+\mathbf{De}((\mathbf{DJ})^+\mathbf{De})^T\right] \\ &= E\left[(\mathbf{DJ})^+\mathbf{De}\ \mathbf{e}^T\mathbf{D}^T((\mathbf{DJ})^+)^T\right] \end{aligned} \tag{4.72}$$

Since the uncertainty lies on $\mathbf{e}$ we have:

$$\begin{aligned} \mathbf{\Sigma}_{\delta\mathbf{r}} &= (\mathbf{DJ})^+\mathbf{D}E\left[\mathbf{e}\mathbf{e}^T\right]\mathbf{D}^T((\mathbf{DJ})^+)^T \\ &= (\mathbf{DJ})^+\mathbf{D}\mathbf{\Sigma}_{\mathbf{e}}\mathbf{D}^T((\mathbf{DJ})^+)^T \end{aligned} \tag{4.73}$$

We can actually determine a covariance matrix for the whole set of the visual features or one for each type, giving three covariance matrices, $\mathbf{\Sigma}^g_{\delta\mathbf{r}}$, $\mathbf{\Sigma}^c_{\delta\mathbf{r}}$, and $\mathbf{\Sigma}^p_{\delta\mathbf{r}}$, with following expressions:

$$\mathbf{\Sigma}^{g}_{\delta\mathbf{r}} = (\mathbf{D}^{g}\mathbf{J}_{\mathbf{e}^{g}})^{+}\mathbf{D}^{g2}((\mathbf{D}^{g}\mathbf{J}_{\mathbf{e}^{g}})^{+})^{T} \tag{4.74}$$

$$\mathbf{\Sigma}^{c}_{\delta\mathbf{r}} = (\mathbf{D}^{c}\mathbf{J}_{\mathbf{e}^{c}})^{+}\mathbf{D}^{c2}((\mathbf{D}^{c}\mathbf{J}_{\mathbf{e}^{c}})^{+})^{T} \tag{4.75}$$

$$\mathbf{\Sigma}^{p}_{\delta\mathbf{r}} = (\mathbf{D}^{p}\mathbf{J}_{\mathbf{e}^{p}})^{+}\mathbf{D}^{p2}((\mathbf{D}^{p}\mathbf{J}_{\mathbf{e}^{p}})^{+})^{T} \tag{4.76}$$

Covariances enable to propagate uncertainty from the low-level visual features (through equation(4.19) and (4.63) for the geometrical edge and keypoint features, and through the covariances computations for the color features) to the high-level estimate of the camera pose. The covariance matrix $\mathbf{\Sigma}_{\delta\mathbf{r}}$ is further used in the derivation Kalman filter presented in the next section.

4.5.2 Kalman filtering and pose prediction

4.5.2.1 Kalman filtering

In order to smooth pose estimates, we propose to incorporate a filtering process, relying on the Kalman filtering theory. To achieve this, we employ an linear Kalman filter on the parameters of the camera velocity $\mathbf{v}$, which is integrated to determine the pose so that: $\mathbf{M}_{k+1} = exp([\mathbf{v}_{k+1}\delta t])\ \mathbf{M}_k$, at each time step k (we use the notation $\mathbf{M}$ instead of ${}^{c}\mathbf{M}_{o}$ for clarity reasons). We assume a constant velocity dynamic model. This model is particularly suitable in our context of space rendezvous since constant velocity motions are generally involved.

As the state of the system $(\widehat{\mathbf{x}}_k, \mathbf{\Sigma}_k)$ we simply choose the velocity parameters, so that $\mathbf{x}_k = \mathbf{v}_k$ is the actual system state at time step k, with $\widehat{\mathbf{x}}_k = \widehat{\mathbf{v}}_k$ the a posteriori estimate of $\mathbf{x}_k$, and $\mathbf{\Sigma}_k$ the corresponding covariance matrix:

$$\widehat{\mathbf{x}}_k = \mathbf{x}_k + \eta_k \tag{4.77}$$

$$\text{and} \quad \widehat{\mathbf{M}}_k = exp([\eta_k\delta t])\mathbf{M}_k \tag{4.78}$$

with $\eta_k \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_k)$, and $\widehat{\mathbf{M}}_k$ the posterior estimate of the actual pose $\mathbf{M}_k$.

With a constant velocity model, the true state at time step $k+1$ is evolved from the state at k according to:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{n}_{k+1} \tag{4.79}$$

with $\mathbf{n}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$ the state noise, and $\mathbf{Q}_k = diagblock(\mathbf{\Sigma}_{\boldsymbol{\upsilon}}, \mathbf{\Sigma}_{\boldsymbol{\omega}})$ the state noise covariance matrix, with $\mathbf{\Sigma}_{\boldsymbol{\upsilon}} = \sigma^2_{\boldsymbol{\upsilon}}\mathbf{I}_{3\times3}$ and $\mathbf{\Sigma}_{\boldsymbol{\omega}} = \sigma^2_{\boldsymbol{\omega}}\mathbf{I}_{3\times3}$, $\sigma_{\boldsymbol{\upsilon}}$ and $\sigma_{\boldsymbol{\omega}}$ being state noise standard deviations respectively on the translation and rotation parameters of $\mathbf{v}$. As an observation $\mathbf{z}_{k+1}$, the minimization process described in section 4.1 (and equation(4.71)) provides us with a pose measure $\mathbf{M}^{m}_{k+1}$, and with a measure of the velocity $\mathbf{v}^{m}_{k+1}$:

$$\mathbf{v}^{m}_{k+1} = exp^{-1}(\widehat{\mathbf{M}}_k\ \mathbf{M}^{m\,-1}_{k+1}) \tag{4.80}$$

$$\mathbf{z}_{k+1} = \mathbf{v}^{m}_{k+1} = \mathbf{x}_{k+1} + \mathbf{w}_{k+1} \tag{4.81}$$

with $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$, with $\mathbf{R}_k$ the observation noise covariance matrix. The noise $\mathbf{w}_k$ can actually be interpreted as equal to $\delta\mathbf{r}_k$, which is the pose error at the end of the minimization process of the pose estimation, so that $\mathbf{R}_k = \mathbf{\Sigma}_{\delta\mathbf{r}}$, the computation of $\mathbf{\Sigma}_{\delta\mathbf{r}}$ being

developed in section 4.5.1. The prediction step can then be achieved, giving prior estimate of future state:

$$\widehat{\mathbf{x}}_{k+1|k} = \widehat{\mathbf{x}}_k \tag{4.82}$$

$$\mathbf{\Sigma}_{k+1|k} = \mathbf{\Sigma}_k + \mathbf{Q}_{k+1} \tag{4.83}$$

Similarly we can obtain a prior pose:

$$\widehat{\mathbf{M}}_{k+1|k} = exp(\left[\widehat{\mathbf{x}}_{k+1|k}\right])\widehat{\mathbf{M}}_k \tag{4.84}$$

The update step is classically performed, resulting in the Kalman gain $\boldsymbol{K}_{k+1}$ and in posterior stated estimates $\widehat{\mathbf{x}}_{k+1}$ and $\mathbf{\Sigma}_{k+1}$:

$$\boldsymbol{K}_{k+1} = \mathbf{\Sigma}_{k+1|k}(\mathbf{\Sigma}_{k+1|k} + \mathbf{R}_k)^{-1} \tag{4.85}$$

$$\widehat{\mathbf{x}}_{k+1} = \widehat{\mathbf{x}}_{k+1|k} + \boldsymbol{K}_{k+1}(\mathbf{z}_{k+1} - \widehat{\mathbf{x}}_{k+1|k}) \tag{4.86}$$

$$\mathbf{\Sigma}_{k+1} = (\mathbf{I} - \boldsymbol{K}_{k+1})\mathbf{\Sigma}_{k+1|k} \tag{4.87}$$

A posterior pose estimate $\widehat{\mathbf{M}}_{k+1}$ can also be determined:

$$\widehat{\mathbf{M}}_{k+1} = exp(\left[\boldsymbol{K}_{k+1}(\mathbf{z}_{k+1} - \widehat{\mathbf{x}}_{k+1|k})\right])\widehat{\mathbf{M}}_{k+1|k} \tag{4.88}$$

4.5.2.2 Pose prediction

From the estimate pose $\widehat{\mathbf{M}}_{k+1}$, we suggest to use it, along with the Kalman filter equations, to provide a predicted pose $\widehat{\mathbf{M}}^p_{k+1}$ which is intended to initialize the pose estimation phase for the next time step. A natural idea is to choose the prior estimate at $k+1$ so that:

$$\widehat{\mathbf{M}}^{pred}_{k+1} = exp([\widehat{\mathbf{x}}_{k+1}])\widehat{\mathbf{M}}_{k+1} \tag{4.89}$$

However such a prediction step can be too harsh when the state noise covariance parameters are not properly tuned. We can also propose to incorporate the Kalman gain $\boldsymbol{K}_{k+1}$ with the objective of making this prediction smoother, as a trade-off between the dynamic model and the previous pose posterior estimate:

$$\widehat{\mathbf{M}}^{pred}_{k+1} = exp([(\mathbf{I} - \boldsymbol{K}_{k+1})\widehat{\mathbf{x}}_{k+1}])\widehat{\mathbf{M}}_{k+1} \tag{4.90}$$

This prediction step will be particularly useful when large inter-frame motions are observed in the image, since the model projection with respect to $\widehat{\mathbf{M}}^{pred}_{k+1}$ can bring the error function Δ closer to its actual minimum, avoiding local ones and avoiding to tune low-level tracking parameters such as 1D ranges for the Moving Edge process or the color features computation, to large but risky values.

4.6 Experimental results

This section provides some results of experiments carried out to evaluate the performance of the solutions proposed in this work to handle the problem of pose estimation via frame-by-frame tracking. These experiments were mainly performed in the case of space rendezvous and proximity operations applications but, as it will be seen, it is not restricted to

such issues. These experiments aim at determining qualitative results, on both synthetic and real images, but also quantitative results on synthetic images since such data can be provided with ground truth (section 4.6.1). A comparative study has also been achieved between the different approaches proposed in this work and with respects to some state-of-the-art methods, emphasizing their respective advantages and drawbacks. The relevance of these approaches, particularly concerning the pose measurement requirements involved in a rendezvous mission, as described in section 1.2, is also shown.

Another line of experiments have intended to prove the feasibility of a GNC framework of a close range rendezvous only based on a monocular camera sensor by applying our pose estimation framework within a closed control loop using visual servoing (section 4.6.2).

4.6.1 Qualitative and quantitative evaluation

In this section we validate the proposed method, both qualitatively on real images and qualitatively on synthetic images in order to verify the benefits of our contributions.

4.6.1.1 Implementation

The rendering process of the 3D polygonal and textured model relies, as for the detection method previously presented, on OpenSceneGraph. As presented in section 4.2, we have considered shader programming for some image processing steps during the rendering and edge generation phases. This is done using OpenGL Shading Language (GLSL), supported by OpenSceneGraph, which enables classic shading techniques such as composition of successive shaders. The remainder of the algorithm has been implemented thanks to the C++ ViSP library [Marchand 05]. Regarding hardware, a standard laptop with an NVIDIA NVS 3100M graphic card has been used, along with a 2.8GHz Intel Core i7 CPU.

4.6.1.2 Results on synthetic images

We have achieved a quantitative evaluation of our algorithm on synthetic images, using the realistic ray-tracing simulator developed by Astrium for space environments, and which has been introduced in section 1.4.2. We present a sequence which features a Spot satellite and which is provided with ground truth.

Spot satellites are earth observation satellites whose orbit is approximately polar, circular, sun-synchronous, at an altitude of around 830 kilometers, and with an inclination of 98.7 degrees (see Figure 4.9). For space debris removal concerns, we consider an arbitrary rotation for the target attitude and a chaser spacecraft is supposed to be located on a similar orbit, with a slightly different eccentricity in order to make the chaser fly around the target, in the $x_{Orb} - z_{Orb}$ plane of the orbital frame (see Figure 4.10). The chaser is also equipped with a virtual camera providing images of the target and a spot light to lighten the target, especially to handle sun eclipses. This sequence allows to evaluate the different solutions under various conditions for the distance range (from 20m to 76m, Figure 4.11), for the illumination conditions (low luminosity on frame 920, Figure 4.13), for noise and

specular effects (on frames 40 and 1410) and for the background (cluttered on frame 40 with the earth, uniform on frame 1270 with deep space).

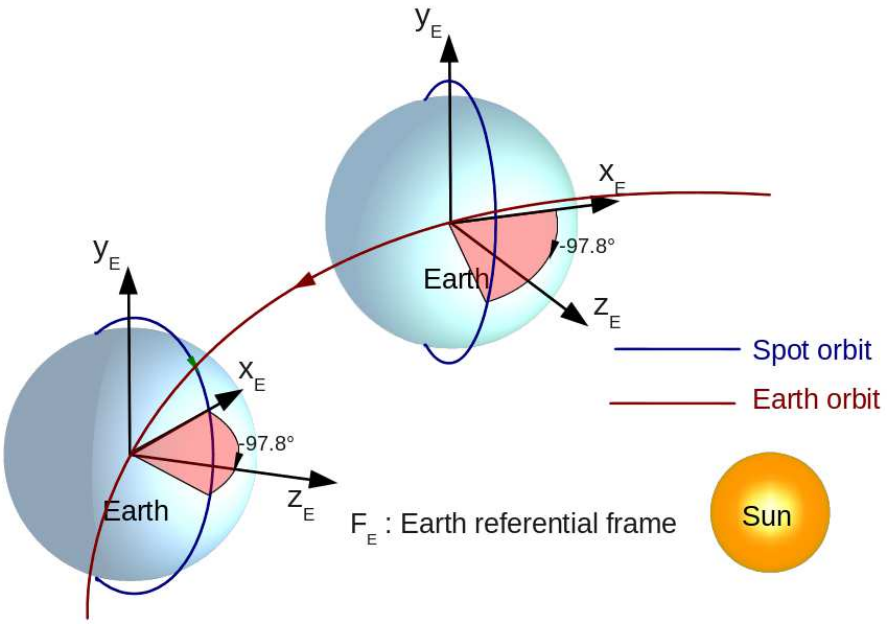
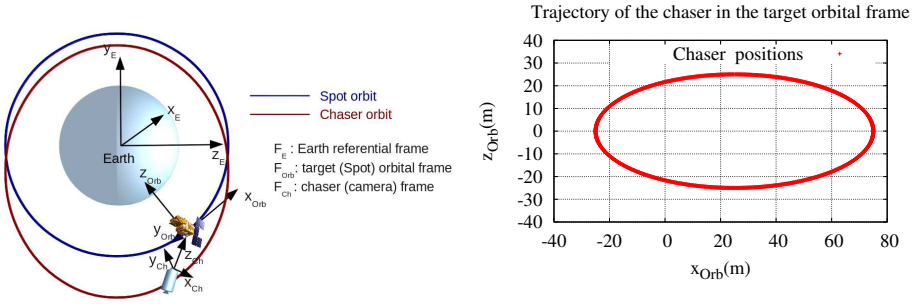


Figure 4.9 – Spot sun-synchronous orbit.

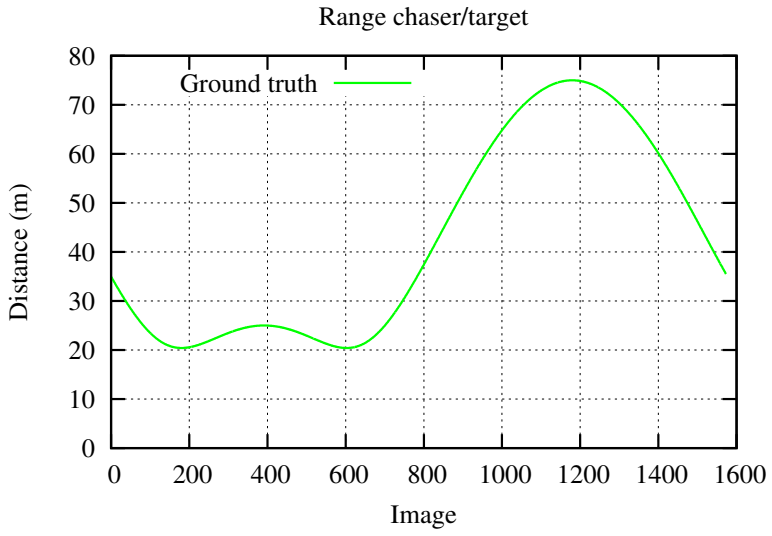


(a) (b)

Figure 4.10 – Chaser and target (Spot) orbits in the Earth reference frame (a). Chaser trajectory in the target local orbital frame (b), whose x_{Orb}-axis is in the target velocity vector direction, with the z_{Orb} axis pointing towards the earth center.

We have investigated the performances of the different solutions designed in this thesis comparatively between them and comparatively to state-of-the art methods such as [Vacchetti 04c, Panin 06, Panin 08b] and [Comport 06a]. Let us first classify the different contributions of this thesis regarding pose estimation by 3D frame-by-frame tracking:

- $C0$: efficient model projection and model edge control point generation, exposed in section 4.2, and pose estimation based on geometrical edge features with a single hypothesis edge registration phase.
- $C1$: integration of a multiple-hypothesis framework in the edge registration process.



(a)

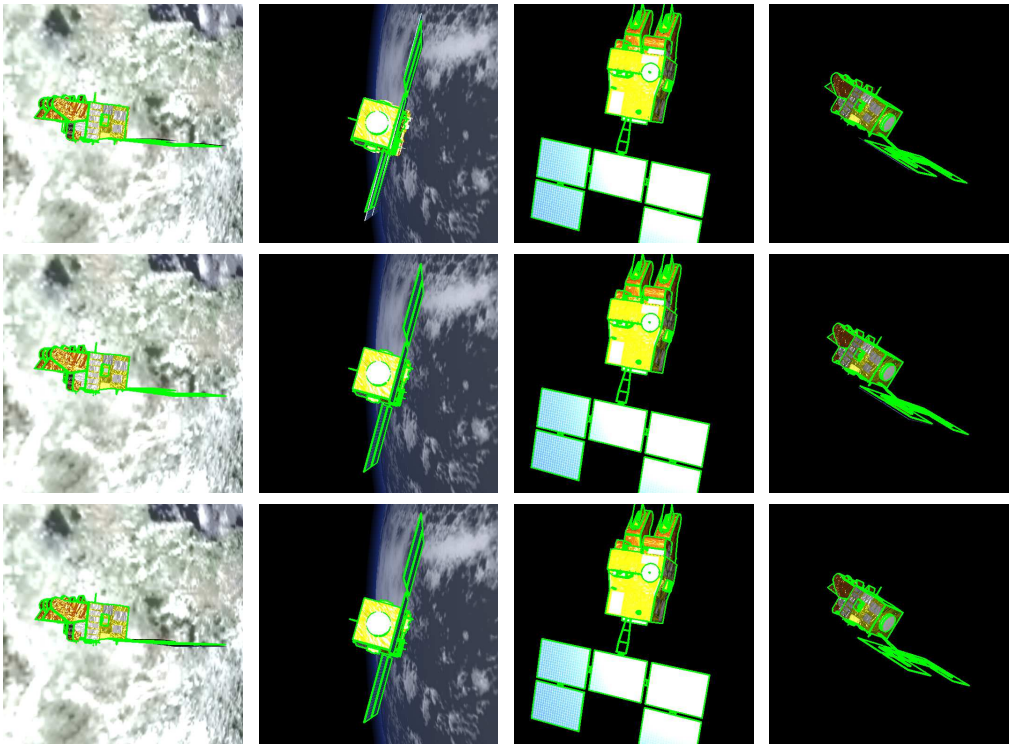
Figure 4.11 – Range between the camera and the target.

- $C2$: incorporating line primitives into our multiple-hypotheses framework for the edge-based registration process, as described in section 4.3.1.3.
- $C3$: integration of the color-based visual features, see section 4.3.2.
- $C4$: temporal consistency for the color-based objective function, presented in section 4.3.2.3.
- $C5$: integration of the geometrical interest points features, introduced in section 4.3.3.
- $C6$: Kalman filtering and pose prediction step (section 4.5).

Solution $C0$ is actually quite close to [Comport 06a] in terms of low-level edge-based tracking and pose estimation minimization process. However, let us remind and point out that the management of the 3D model projection is very different and that [Comport 06a], based on lines of polygonal model, would actually be inapplicable with complex targets. We also propose to compare our solutions with the approaches suggested by [Panin 06], which is equivalent to solution [$C3$], and by [Panin 08b]. For [Panin 08b] we have not implemented the algorithm exactly the same way as in the paper. Instead we have tested a solution fusing the edge-based and color-based objective functions, without M-estimators, without the multiple-hypotheses frameworks and without the temporal consistency for the color-based criteria. It is equivalent to combining $C0$ (but without M-estimators) with $C3$. Let us finally note that combining $C0$ with $C1$ and $C5$ actually makes up an approach similar to [Vacchetti 04b], although [Vacchetti 04b] suffers from the same problem as [Comport 06a] regarding model projection and 2D-2D point correspondences are processed instead of 3D-2D correspondences in our case for the interest points registration phase.

Results for single cue approaches

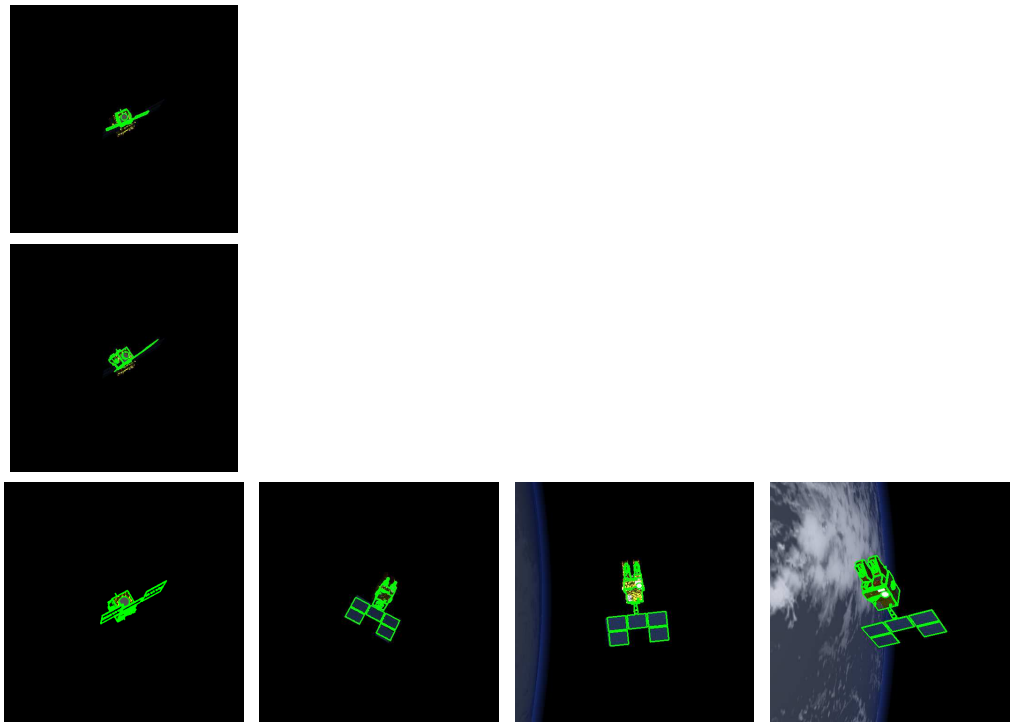
The first set of experiments with this sequence concerns the comparison of single cue approaches. More formally, we compare the solutions provided by $[C0]$, $[C0, C1]$, $[C0, C1, C2]$, and by $[C3]$. The results can be seen on Figure 4.14 where the accuracy of rotation and translation components of an estimated camera pose $\widehat{\mathbf{r}}$ with respects to the true pose $\mathbf{r}^*$ is determined throughout the sequence, through error plots on the pose parameters. Figures 4.12, 4.13 qualitatively show the performances of $[C0]$, $[C0, C1]$ and $[C3]$, with the reprojection of the processed edge control points depicted in green in the image. Tracking failures can be observed for $[C3]$ and $[C0]$ respectively around frames 920, as confirmed by the error plots.



frame 40 frame 270 frame 550 frame 750

Figure 4.12 – Results, from top to bottom with $[C0]$, $[C3]$ and $[C0, C1]$, frames 40-750

We notice that among the single cue solutions, only $[C0, C1]$, which consists in the geometrical edge-based criteria along with a multiple hypothesis framework is able to track and estimate the pose of the camera with respect to the target object throughout the sequence. With simply $[C0]$ or $[C3]$, tracking fails around frame 900 when the target is getting far, with low luminosity (as depicted with frame 920 on Figure 4.13), resulting in noisy edges and in a poor color contrast. The absence of multiple-hypothesis for $[C0]$ causes the numerous outliers encountered in these conditions not to be rejected. For $[C3]$, the contrast being low, the separation of colors between both of the object silhouette contour cannot be properly carried out. We can finally note on Figure 4.14 that the incorporation of line clustering $[C2]$ for the multiple-hypothesis framework slightly improves tracking performances.



frame 920 frame 1270 frame 1410 frame 1500

Figure 4.13 – Results, from top to bottom with [$C0$], [$C3$] and [$C0, C1$], frames 920-1500.

Results for hybrid approaches

Here we suggest to compare solutions fusing the different sorts of visual features proposed in this work ($C0$, $C3$ and $C5$), along with the different other contributions ($C1$, $C2$, $C4$ and $C6$).
The results presented on Figure 4.17 show that the combination [$C0, C3$], (without the use of M-estimators), what is similar to the solution proposed in [Panin 08b], is not sufficient and the tracking fails around frame 850, as also depicted on Figure 4.15, 4.16). However performances until failure are better, in terms of accuracy, than simply [$C0$] or [$C3$] or than [$C0, C1$], color features providing more accuracy in quite neat conditions. With the addition of multiple hypotheses solutions ($C1$, $C2$) and M-estimators for both geometrical and color-based criteria (solution [$C0, C1, C3$]), tracking is properly performed until the end of the sequence (Figures 4.15, 4.16, 4.17).

We besides observe that the approach combining geometrical edge and interest points features [$C0, C1, C5$], similar to [Vacchetti 04b], also succeeds to track to object. It tends to improve performances on certain phases (frames 1300-1500) with respect to [$C0, C1$] but can also give poorer results (frames 700-900) due to the presence of too few interest points under low luminosity and due to the relative drift resulting from the constraint induced by the tracking of the points. Finally, the solution employing [$C0, C1, C3, C5$] (Figures 4.15, 4.16, 4.17), combining all the different visual features shows, as expected, the best performances and appears to be the most robust to the different conditions involved in this sequence, especially when the satellite is far, with low luminosity (between frame 1200 and 1500). Let us note that since contribution $C2$ requires many model edge control points to be efficient ($N_g = 2000$ in these experiments), hence a quite heavy computational burden, and since it provides only few improvements in the tracking per-

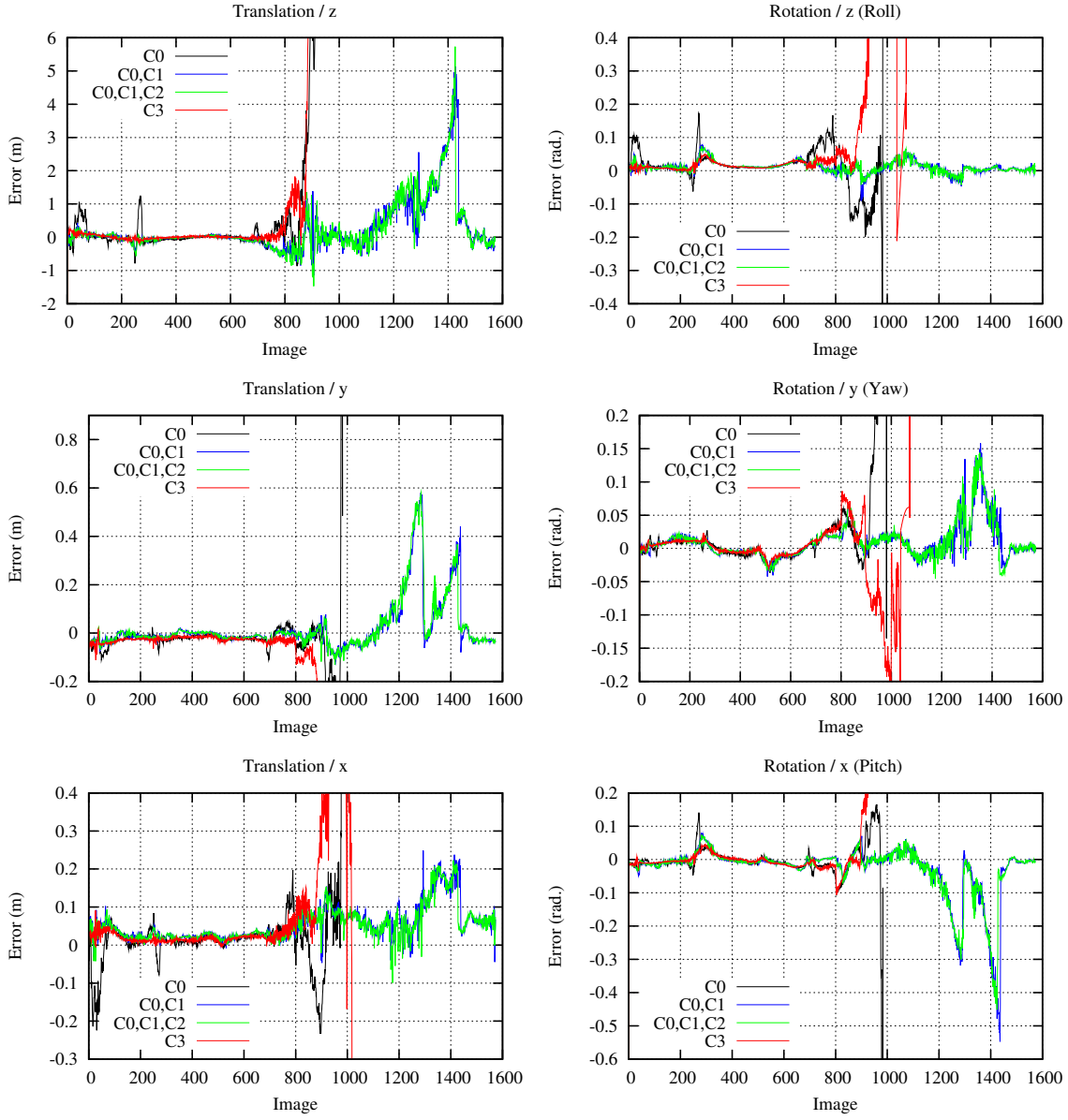
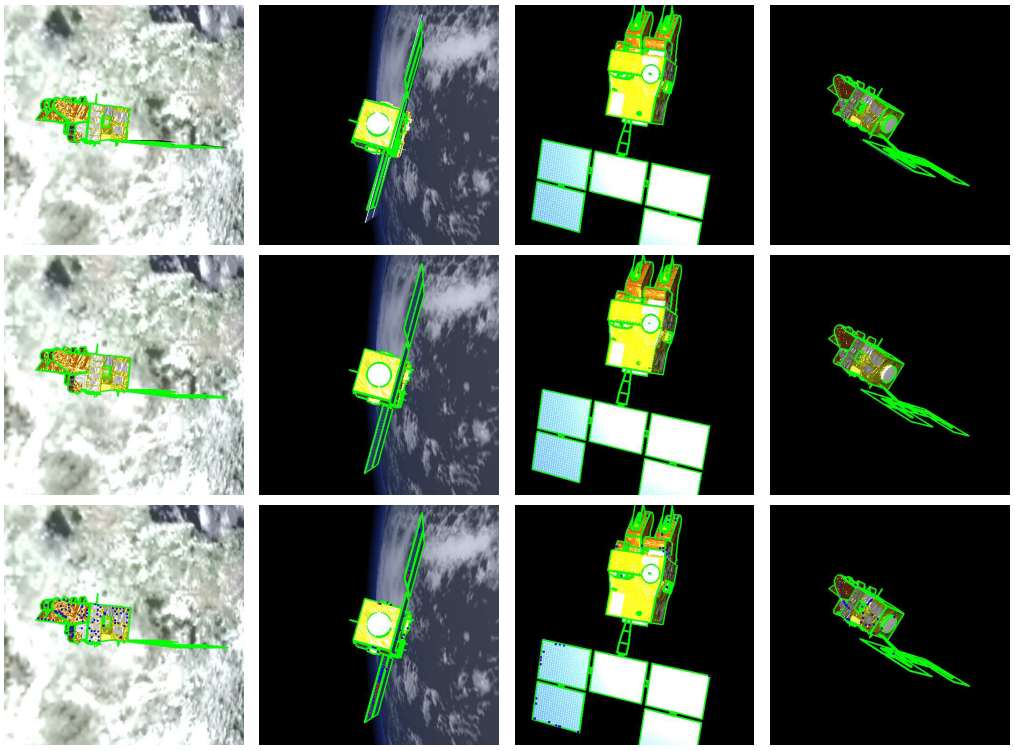


Figure 4.14 – Pose errors on the Soyuz sequence, with solutions [$C0$], [$C0, C1$], [$C0, C1, C2$] and [$C3$].

formances, next tests on hybrid solutions have been carried out without this process. Root mean square errors on the pose parameters are also represented on Table 4.1.

Table 4.1 – RMS erros, for the whole sequence. t_x, t_y, t_z (in meters) and R_x, R_y, R_z (in radians) respectively refer to translation and rotation (Euler angles) parameters.

Mode	t_x	t_y	t_z	R_x	R_y	R_z
C0,C1,C2	0.069	0.109	0.739	0.075	0.029	0.012
C0,C1,C3	0.055	0.046	0.401	0.015	0.021	0.021
C0,C1,C5	0.069	0.147	0.810	0.083	0.031	0.029
C0, C1, C3, C5	0.054	0.045	0.373	0.015	0.020	0.016



frame 40 frame 270 frame 550 frame 750

Figure 4.15 – Results, from top to bottom with [$C0, C3$], [$C0, C1, C3, C4$] and [$C0, C1, C3, C4, C5$], frames 40-750. Blue and red dots for [$C0, C1, C3, C4, C5$] respectively represent tracked and forward-projected KLT points.

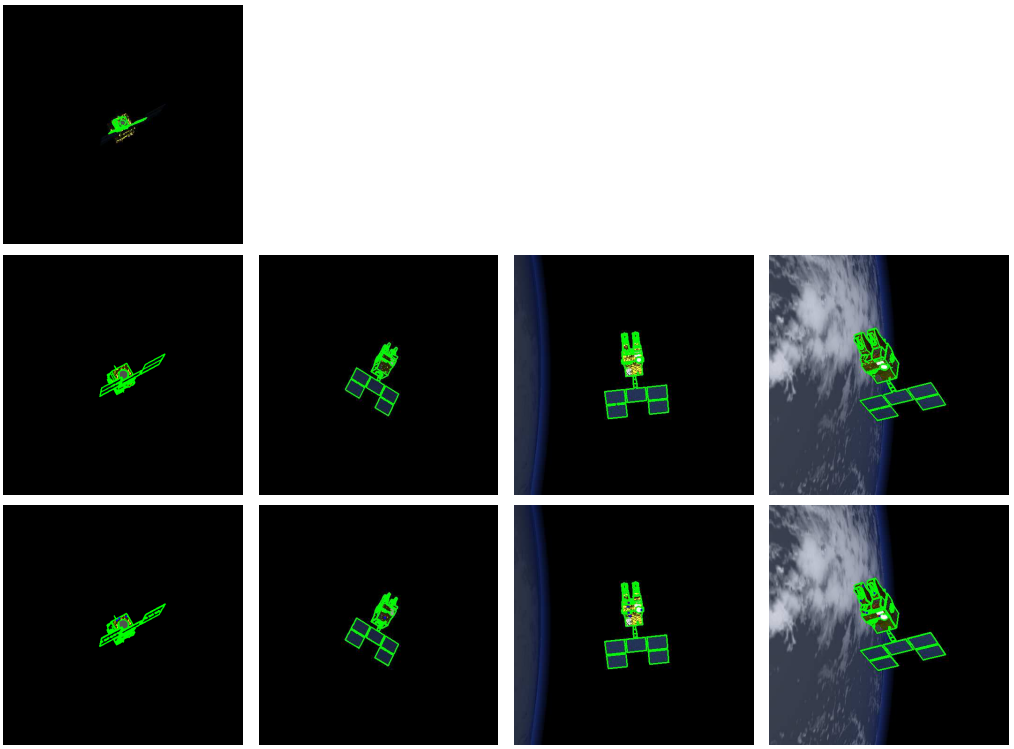


Figure 4.16 – Results, from top to bottom with [$C0, C3$], [$C0, C1, C3$] and [$C0, C1, C3, C5$], frames 890-1500. [$C0, C3$] fails around frame 850-900, whereas [$C0, C1, C3$] and [$C0, C1, C3, C5$] successfully track the object throughout the sequence.

Compliance with close range navigation requirements for space proximity operations

Relating these different performances (Figures 4.14 4.17) with the space rendezvous navigation requirements specified in section 1.2 (from [Fehse 08, Astrium 10]), we can notice that errors for lateral position measurements for [$C0, C1, C3$] and [$C0, C1, C3, C5$] (along t_x and t_y), almost remain under $0.10m$ for the whole sequence, except around frames 800 and 1300 for which conditions are challenging. $0.10m$ is the requirement for lateral misalignment at contact for a docking-based rendezvous mission (Table 1.1). For [$C0, C1$] and [$C0, C1, C2$] lateral errors can be maintained below $0.50m$ (except around frame 1250), which is the performance required for the berthing maneuver at "contact". By plotting the relative position errors with respect to the range (Figure 4.18), we can observe that the accuracy for lateral position measurements of 1% of the range (in the case of a closed loop maneuver) can be clearly achieved by [$C0, C1, C3$], [$C0, C1, C3, C5$], and [$C0, C1, C2$] (and [$C0, C1$] which is very similar), and almost by [$C0, C1, C5$].

In terms of angular misalignment, the requirements (at contact) given on Table 1.1 for docking can vary between $1°$ and $5°$ (or 0.0174 rad. and 0.087 rad.). The upper bound can be almost respected throughout the sequence by [$C0, C1, C3$] and [$C0, C1, C3, C5$]. For berthing, [$C0, C1, C3$] and [$C0, C1, C3, C5$] clearly fulfill the requirements ($10°$).

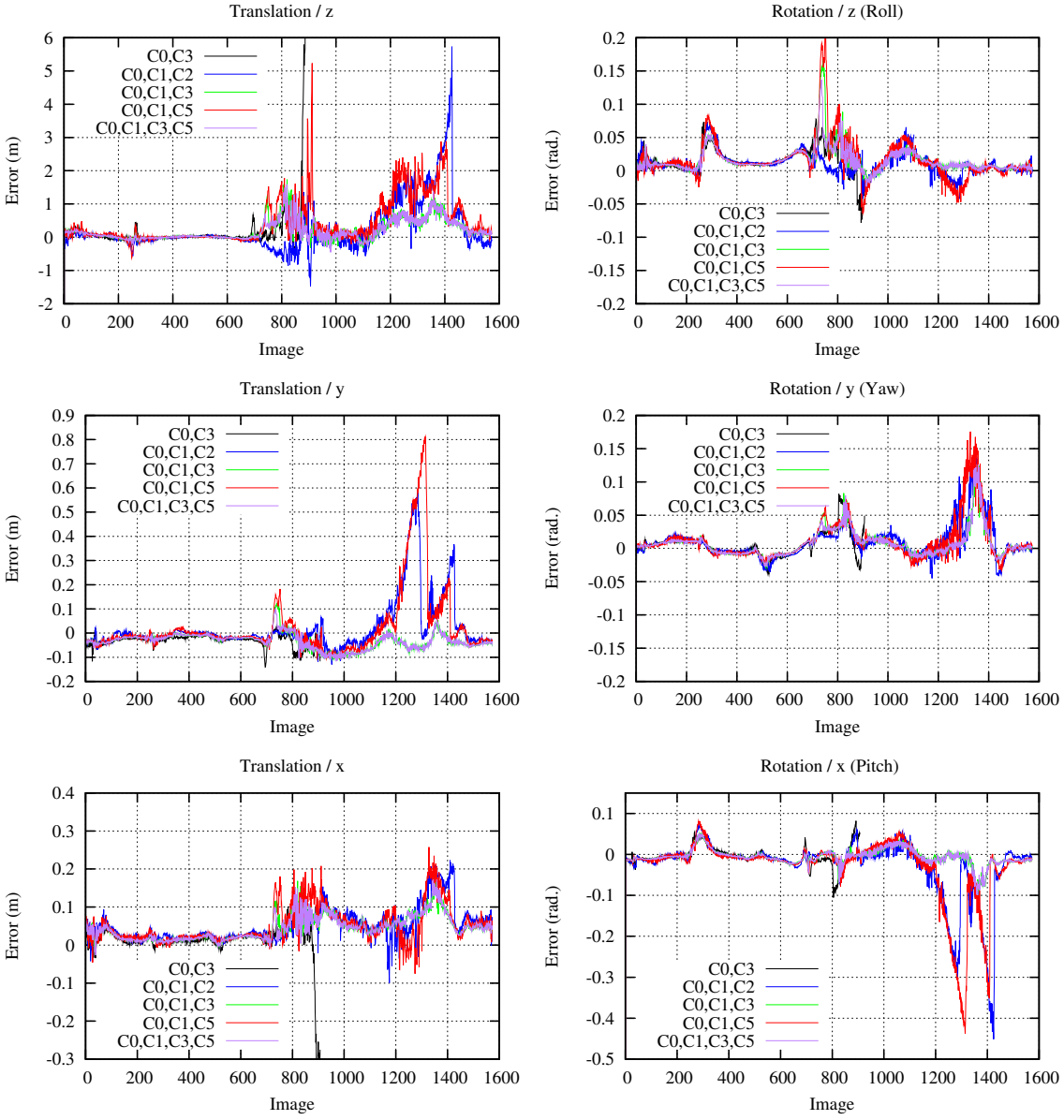


Figure 4.17 – Pose errors on the Spot sequence, with solutions [$C0, C3$], [$C0, C1, C2$], [$C0, C1, C3$], [$C0, C1, C5$] and [$C0, C1, C3, C5$].

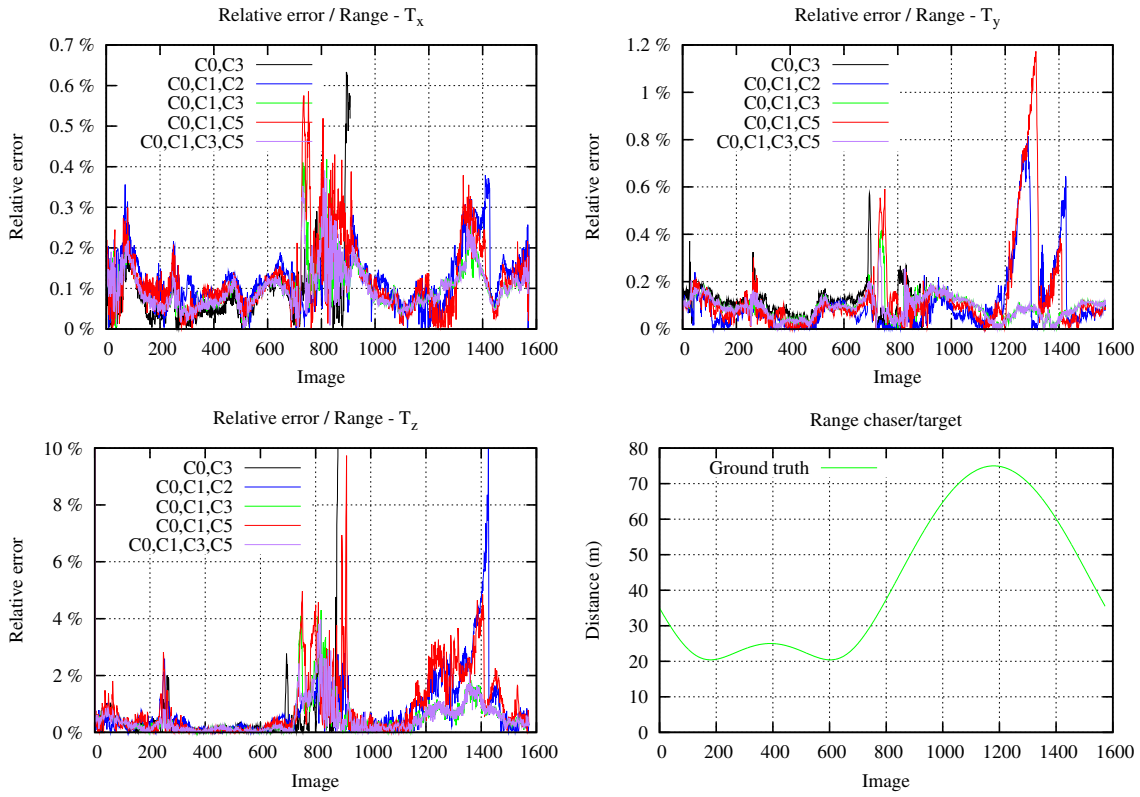


Figure 4.18 – Relative translation errors with respect to the range, on the Spot sequence, with solutions [$C0, C3$], [$C0, C1, C2$], [$C0, C1, C3$], [$C0, C1, C5$] and [$C0, C1, C3, C5$].

Robustness to inter-frame motions

In order to emphasize the efficiency of the hybrid approaches designed in this work, especially with respect to the inter-frame motion of the target in the image, we have downsampled the image sequence by a factor f, meaning that the object appears to move f times faster. With $f = 3$, all the single cue solutions, including $[C0, C1, C2]$, quickly fail (around frame 10), and so are the hybrid solutions $[C0, C1, C5]$ and $[C0, C3]$ (around frame 50). However the other hybrid approaches are still able to properly track the target, even with $f = 5$, as depicted on Figure 4.19. We can notice that the addition of $C4$, which imposes a temporal constraint on color-based error function, slightly improves and smooths results, especially around frame 10, when the solar panels of the satellite flip in the image and around frame 150, when the satellite gets far, with low luminosity (around frame 150). Finally, the incorporation of interest points ($C5$) enables lower errors around frame 150 but some peaks can be observed around frames 50 and 170, when the solar panels flip, due to the fact that interest points detected on the panels failed to be tracked in this case (harsh appearance changes, imprecision of the back-projection). Let us note that $N_g = 500$ model edge control points (resulting in around 250 silhouette edge points) were processed in these tests, and that the scanning ranges R and L (defined in sections 4.3.1 and 4.3.2) for both the geometrical edge features and the color features are set to 14 (the same as the previous experiments).

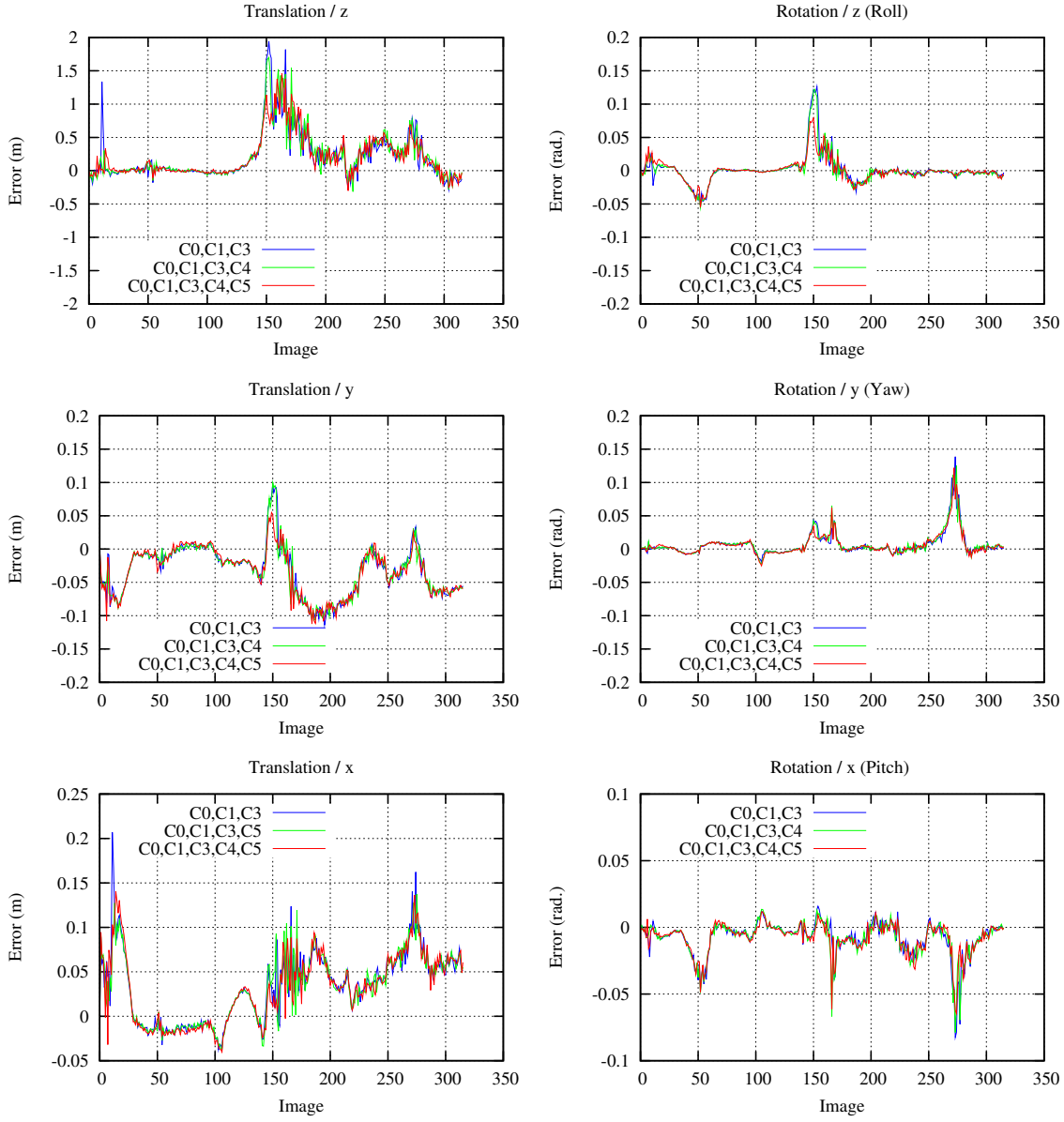


Figure 4.19 – Pose errors on the Spot sequence down-sampled by a factor $f = 5$, with solutions [$C0, C1, C3$] and [$C0, C1, C3, C4$], emphasizing the advantage of $C4$.

Influence of some low-level parameters

By measuring the effect of N_g and L and R, results shown on Figure 4.20, still generated with $f = 5$, emphasize the observations made above. As expected, with $L = D = R = 10$, performances are degraded, tracking being lost with $[C0, C1, C3]$. The addition of $C4$ avoids the failure but some large errors can still be observed between frame 120 and 140. With $N_g = 1000$ and $L = D = R = 14$, the benefit of $C4$ can also be remarked.

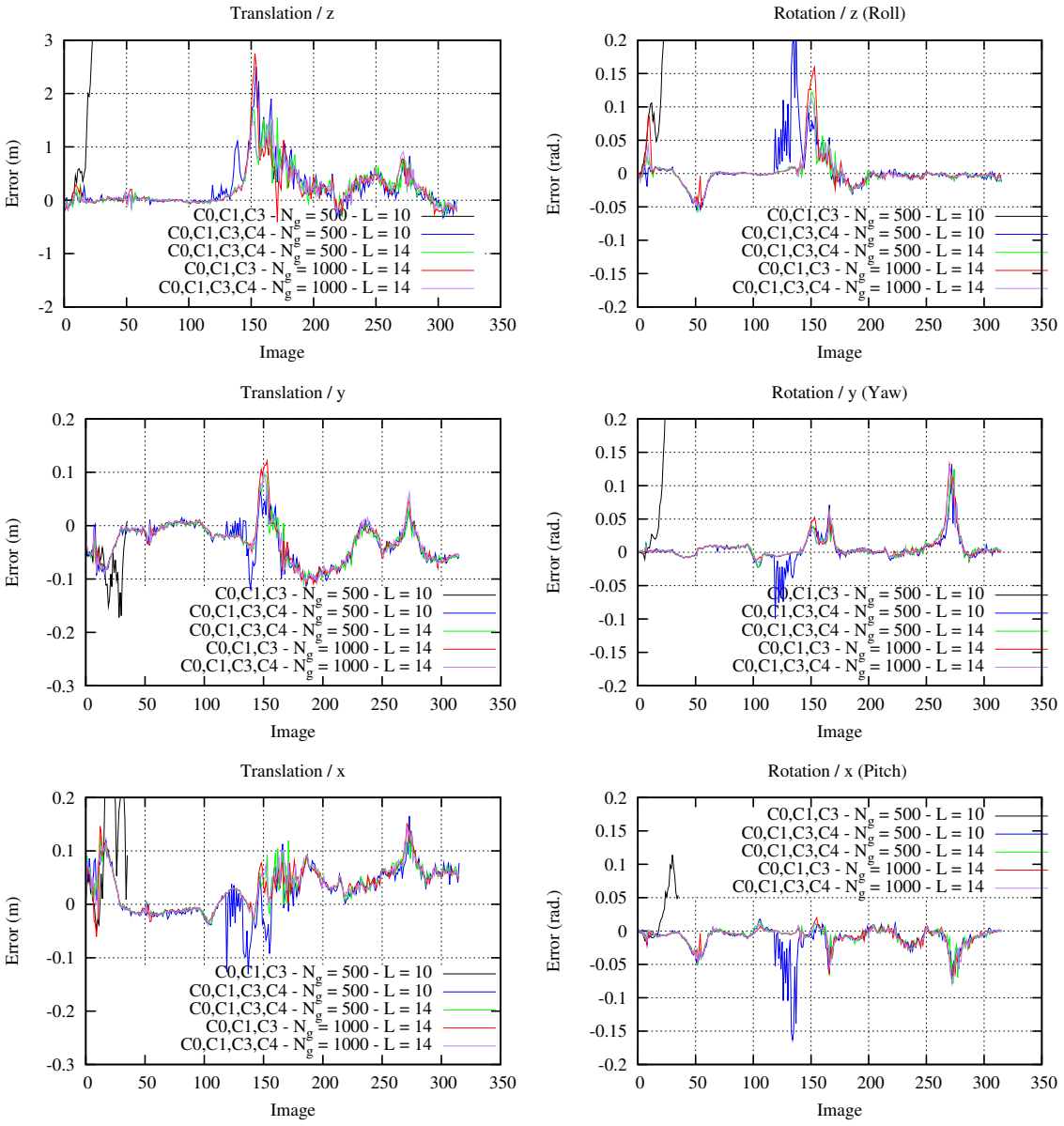


Figure 4.20 – Pose errors on the Spot sequence down-sampled by a factor $f = 5$, with solutions $[C0, C1, C3]$ and $[C0, C1, C3, C4]$, using different tuning parameters: number N_g of edge control points considered and ranges R and L respectively for the geometrical edge feature and the color features.

Kalman filtering

The advantage of the Kalman filtering technique is now examined. Figure 4.21 shows tracking errors for the whole sequence, for the complete solution $[C0, C1, C3, C4, C5]$, along with its filtered version $[C0, C1, C3, C4, C5, C6]$. As expected, errors are smoothed and some peaks are avoided. Let us note that the prediction step is here based on equation (4.90). $N_g = 500$, $L = D = R = 14$ in these experiments.

The benefit of the filter and the prediction are even more stressed out on Figure 4.22 which depicts the tracking performances for the sequence down-sampled with $f = 7$, which implies very large inter-frame motions (up to 25 pixels), making previous tested solutions fail. However through filtering and prediction, tracking can be handled correctly for $[C0, C1, C3, C4, C6]$ and $[C0, C1, C3, C4, C5, C6]$. Prediction is here based on equation (4.89), resulting in a harsher prediction than (4.90) to cope with these large motions. Though enabling a correct initialization from frame-to-frame, this prediction scheme can be sensitive to poor posterior estimate of the velocity parameters from the filter, and thus requires a finer tuning of the state noise parameters than (4.90). $N_g = 500$, $L = D = R = 14$ in these experiments.

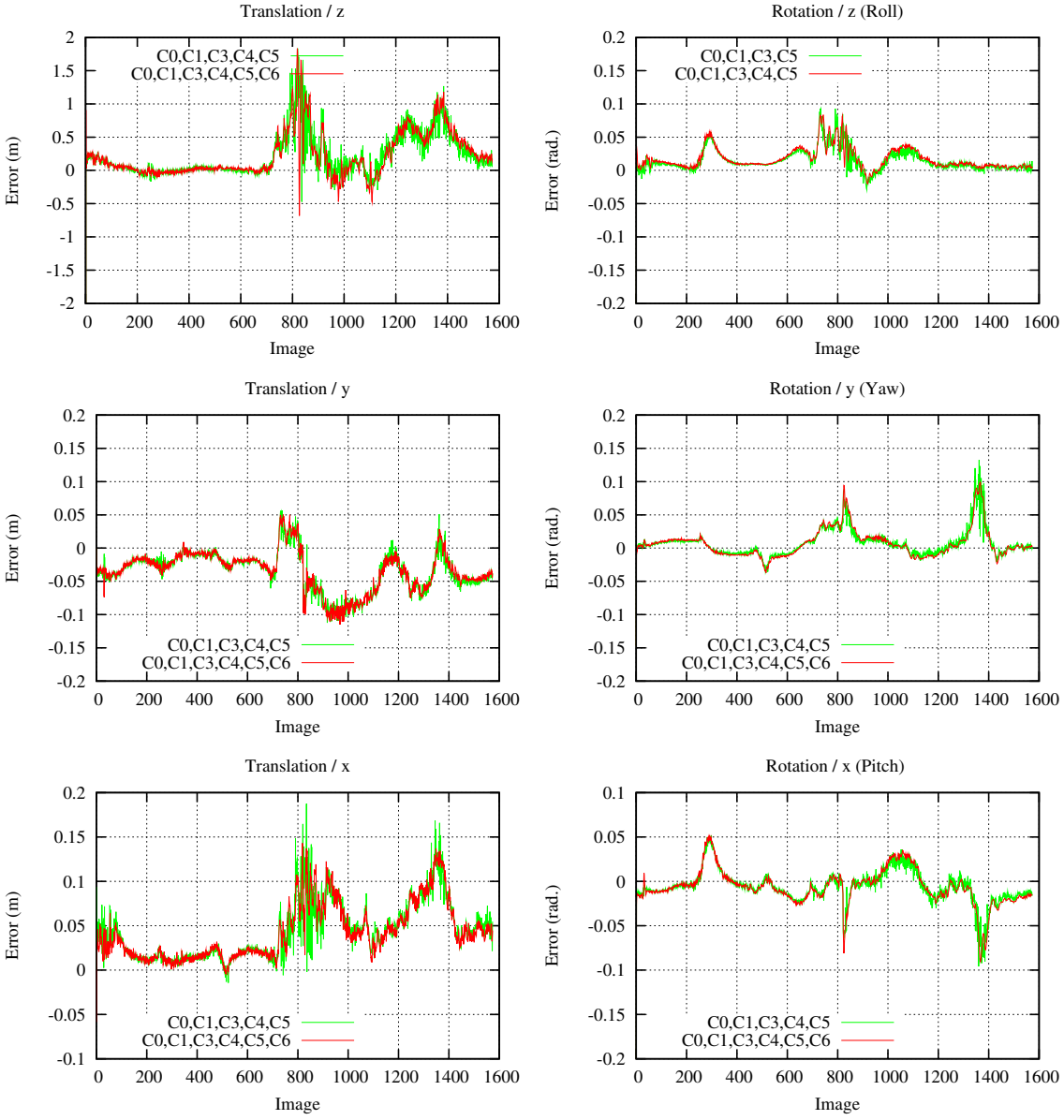


Figure 4.21 – Pose errors on the Spot sequence, with solutions [$C0, C1, C3, C4, C5$] and [$C0, C1, C3, C4, C5, C6$], showing the contribution of the Kalman filter.

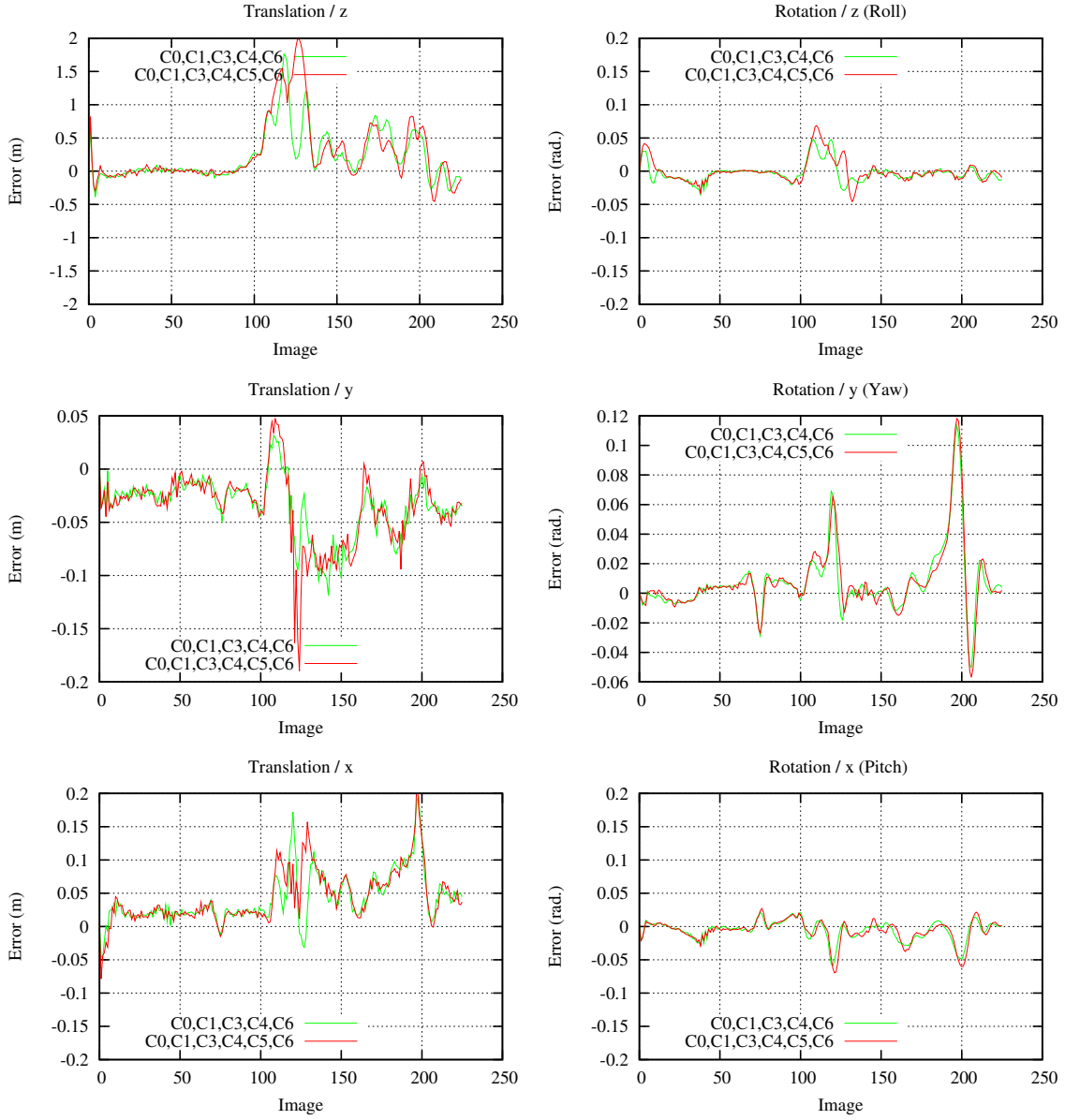


Figure 4.22 – Pose errors on the Spot sequence down-sampled by a factor $f = 7$, with solutions [$C0, C1, C3, C4, C6$] and [$C0, C1, C3, C4, C5, C6$], tracking failing without the prediction step based on the Kalman filter.

4.6.1.3 Results on real images

Soyuz sequence

This sequence shows the Soyuz TMA-03M undocking from the International Space Station (ISS). It can be found on Youtube[1]. We have directly used a descent complete 3D model available on Google 3DWarehouse[2].

We have run on this sequence single cue solutions $[C0]$, $[C0, C1]$ and $[C3]$ (see Figure 4.23). $[C0]$ shows poor performances until the tracking can be qualitatively considered as lost on frame 400-450. $[C0, C1]$ enable improvements, until frames 400-450 when it also diverges. $[C3]$ achieves tracking quite correctly until approximately frame 1100, despite misalignment, as seen on frame 750 for instance. The advantage, on this sequence, of the color-based cue with respect to the geometrical edge-based one can be explained by the noise observed on the internal edges of the object, which degrades performances of $[C0]$, and by the background which progressively turns into deep space, favoring the object/background separation and consequently $[C3]$.

Figure 4.24 shows tracking performing for hybrid solutions $[C0, C3]$, $[C0, C1, C3, C4]$ and $[C0, C1, C3, C4, C5]$. $[C0, C3]$ shows poorer performances than simply $[C3]$, for the reason presented before. However, by adding multiple-hypotheses for edge and temporal constraint for the color cue $[C0, C1, C3, C4]$, we observe that the tracking is properly performed until frame 1700. Results are improved with $[C0, C1, C3, C4, C5]$, with proper tracking until frame 2000. By down-sampling the sequence with $f = 5$, the advantage of introducing interest points is even more obvious since tracking with $[C0, C1, C3, C4, C5]$, which not affected by the increased frame rate (see Figure 4.25), in contrast to $[C0, C1, C3, C4]$ which rapidly fails. The ability of interest points of being properly tracked with large inter-frame motions (up to 25 pixels) is here stressed out.

For the same down-sampled sequence, the uncertainty of the pose for both $[C0, C1, C3, C4]$ and $[C0, C1, C3, C4, C5]$ methods is represented on Figure 4.26 by the global covariance matrix $\mathbf{\Sigma}_{\delta \mathbf{r}}$, which as been introduced in section 4.5.1 as a tool to measure the reliability of the tracking process. On the left is depicted the uncertainty on the translation parameters, by plotting:

$$\sigma_{\boldsymbol{v}} = \sqrt{\mathbf{\Sigma}_{\delta \mathbf{r}}(0,0) + \mathbf{\Sigma}_{\delta \mathbf{r}}(1,1) + \mathbf{\Sigma}_{\delta \mathbf{r}}(2,2)} \tag{4.91}$$

Similarly, the figure on the right shows the uncertainty on the rotation parameters, with the plot of:

$$\sigma_{\boldsymbol{\omega}} = \sqrt{\mathbf{\Sigma}_{\delta \mathbf{r}}(3,3) + \mathbf{\Sigma}_{\delta \mathbf{r}}(4,4) + \mathbf{\Sigma}_{\delta \mathbf{r}}(5,5)} \tag{4.92}$$

We can point out a larger uncertainty when both solutions tend to fail (around frame 20 for $[C0, C1, C3, C4]$ and around frame 420 for $[C0, C1, C3, C4, C5]$).

[1] `http://youtu.be/MlRmTgsDYjk`

[2] `http://sketchup.google.com/3dwarehouse/details?mid=6eb4c556c1f836c3567d8125dd72cc4e`

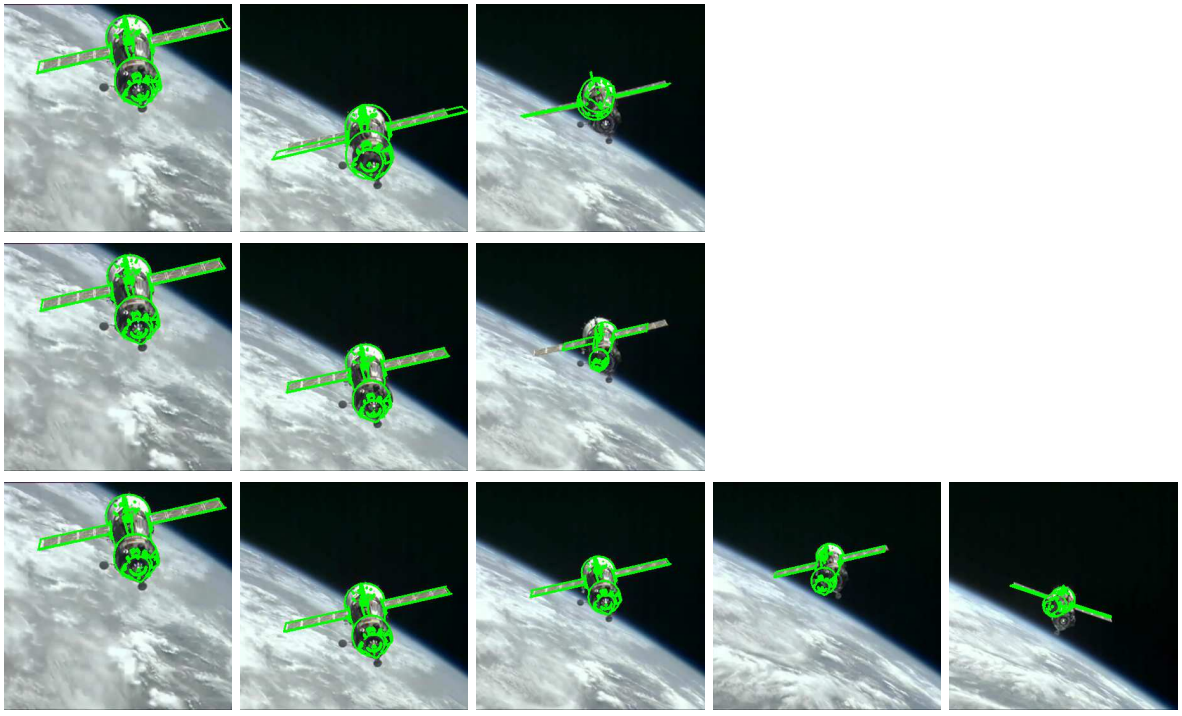


Figure 4.23 – Tracking for the Soyuz sequence with [$C0$] (top), [$C0, C1$] (middle), [$C3$] (bottom).

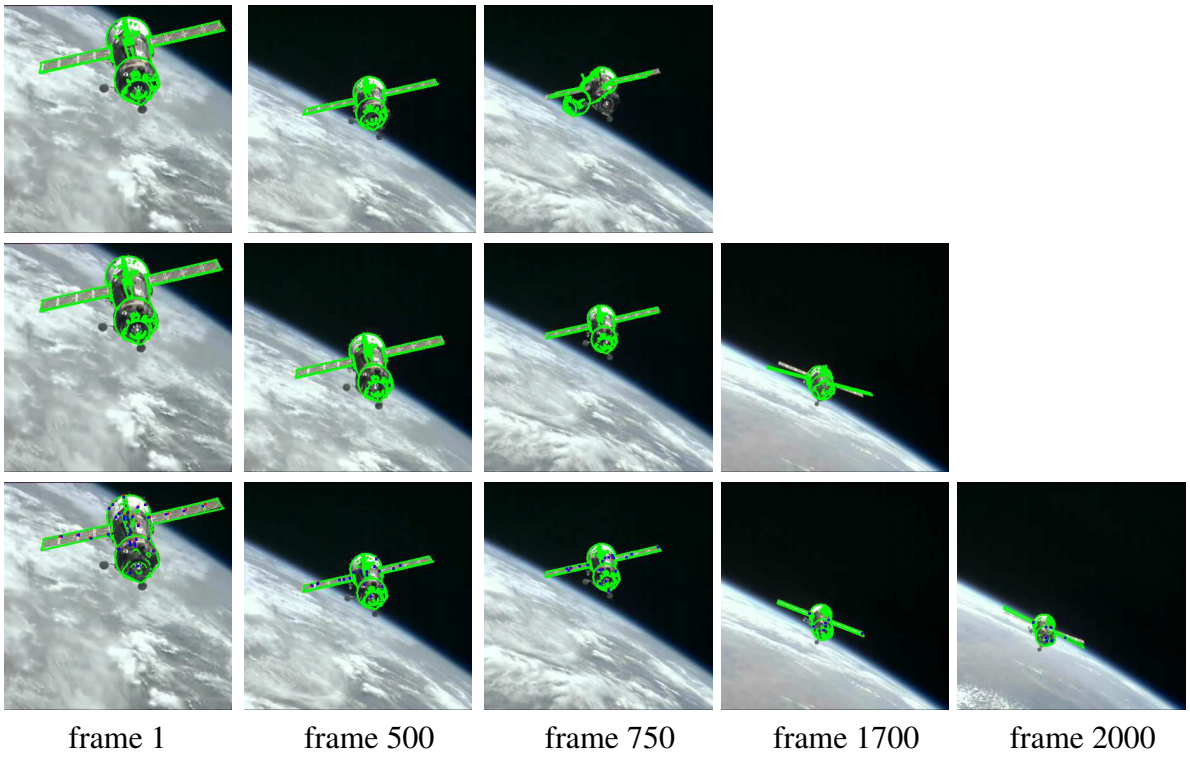


Figure 4.24 – Tracking for the Soyuz sequence with [$C0, C3$] (top), [$C0, C1, C3, C4$] (middle), [$C0, C1, C3, C4, C5$] (bottom).

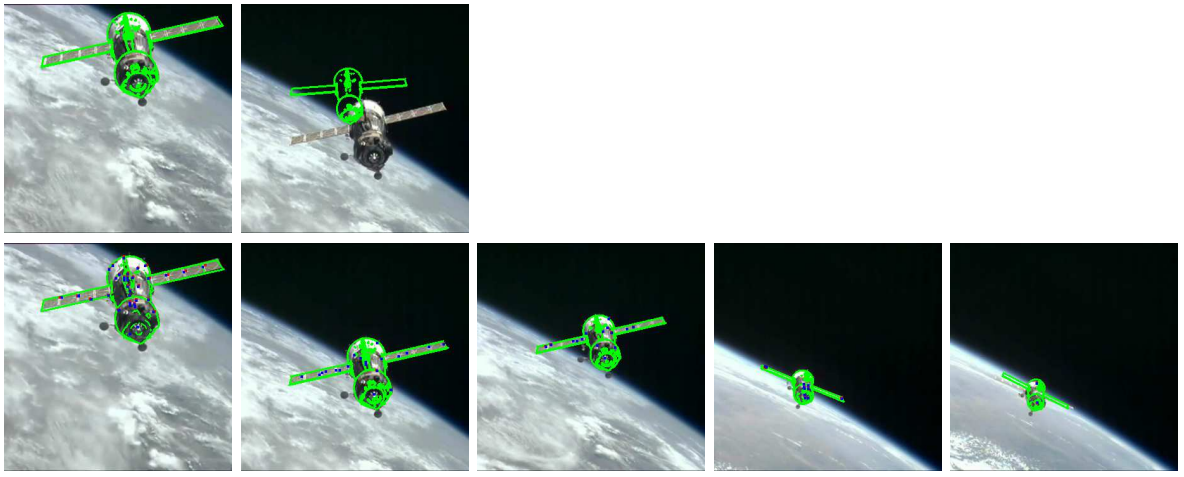


Figure 4.25 – Tracking for the Soyuz sequence with $[C0, C1, C3, C4]$ (top) and with $[C0, C1, C3, C4, C5]$ (bottom), for a down-sampling factor $f = 5$.

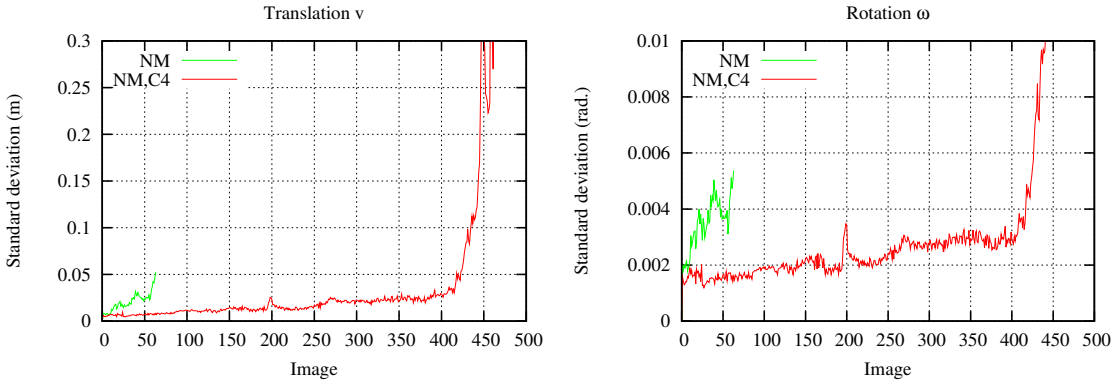


Figure 4.26 – Covariances on the pose errors for $[C0, C1, C3, C4]$ and $[C0, C1, C3, C4, C5]$. $\mathbf{\Sigma}_{\delta \mathbf{r}}$ is represented.

Atlantis sequence

A second example concerns the tracking of the Atlantis Space shuttle, performing a pitch maneuver as it rendezvous with the ISS, prior to docking, for the STS-135 mission. The investigated sequence can also be found on Youtube[3] . An untextured 3D model of the spacecraft has been directly processed for the tracking (available on Google 3DWarehouse[4]). This target also presents a complex shape, with curved parts, such as the fuselage or the engines.
Figure 4.28 shows the tracking performed over the sequence with $[C0, C1]$ (row 1), $[C0, C1, C3, C4]$ (row 2), $[C0, C1, C3, C4, C5]$ (row 3), $[C0, C1, C3, C4, C5, C6]$ (row 4), the addiction of the successive contributions making the tracking robust to some illumination changes, to some large motions (thanks to $C5$ around frame 80) and challenging situations when the shuttle flips in the image (around frame 660 and 2540 thanks to $C3$ and $C6$). In the same manner as for the Soyuz sequence, the covariance parameters are also represented on Figure 4.29 for the different considered tracking solutions. The successive failures of $[C0, C1]$, $[C0, C1, C3, C4]$ and $[C0, C0, C1, C3, C5]$ can be respectively observed around frames 650, 2700 and 2800 with growing covariance parameters.

[3] http://youtu.be/ZYb0p991x1Y
[4] http://sketchup.google.com/3dwarehouse/details?mid=a46d59be3ac09ac4843ecb708acc7f22&prevstart=0

frame 8 frame 80 frame 335

Figure 4.27 – Tracking for the Atlantis sequence with, from top to bottom, [$C0, C1$], [$C0, C1, C3, C4$], [$C0, C1, C3, C4, C5$] and [$C0, C1, C3, C4, C5, C6$], frames 8-335

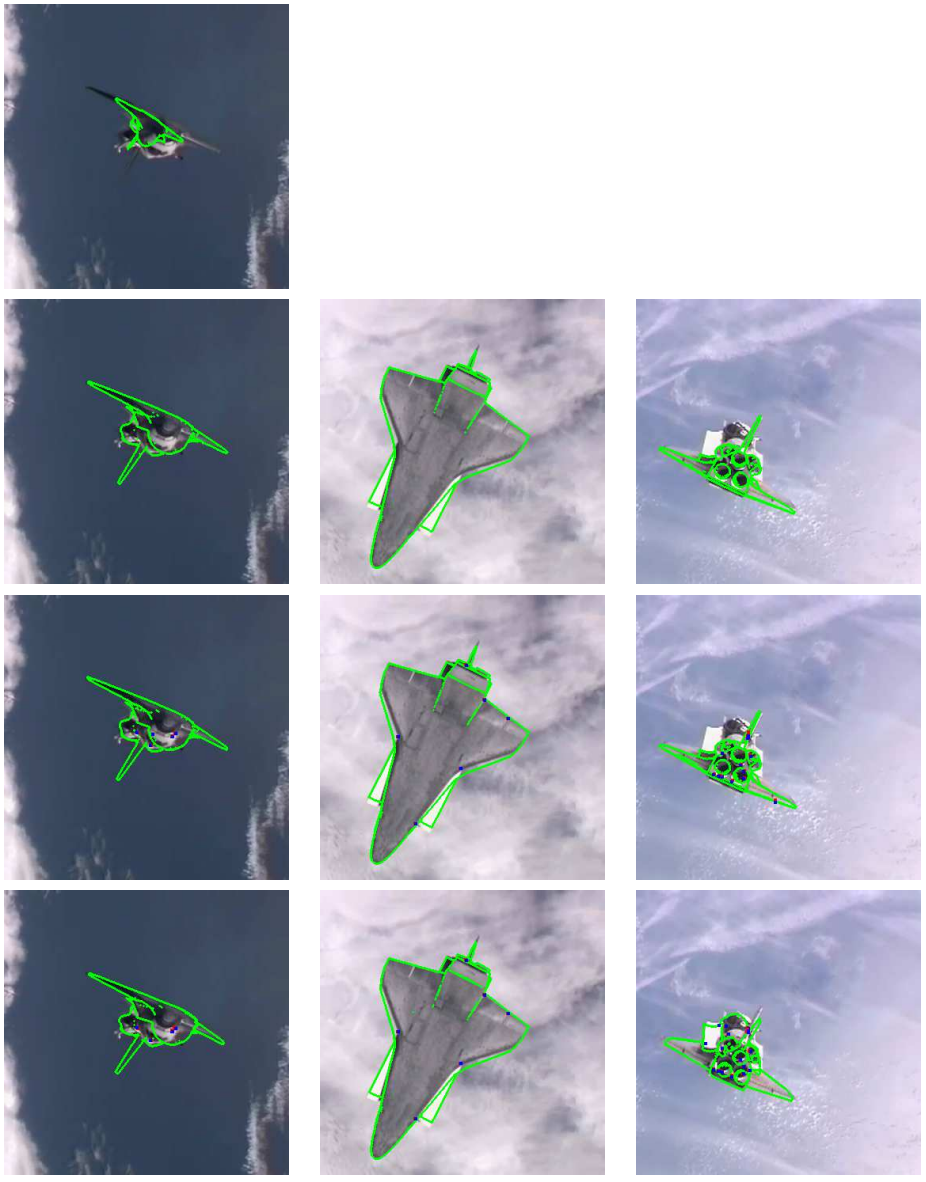


Figure 4.28 – Tracking for the Atlantis sequence with, from top to bottom, [$C0, C1$], [$C0, C1, C3, C4$], [$C0, C1, C3, C4, C5$] and [$C0, C1, C3, C4, C5, C6$], frames 660-2540.

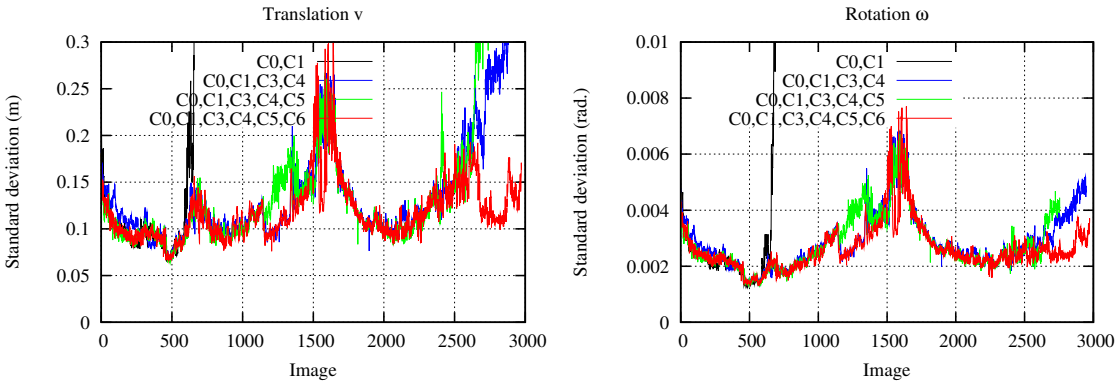


Figure 4.29 – Covariances on the pose errors, for $[C0, C1]$, $[C0, C1, C3, C4]$, $[C0, C1, C3, C4, C5]$ and $[C0, C1, C3, C4, C5, C6]$. Square root of traces on translation and rotation parameters of $\Sigma_{\delta \mathbf{r}}$ are represented.

Mock-ups video sequences

Two sequences involving mock-ups of satellites are processed.

Amazonas sequence: the first one has been taken using the Lagadic robotic platform, presented in section 1.4, and the 1/50 mock-up of Amazonas-2, provided by Astrium (section 1.4). The six degrees of freedom robot has been used to simulate a space rendezvous, with a camera mounted on the end-effector of the robot, and enables to have regular and quite realistic movements. Let us however remind that the specific dynamic of the chaser spacecraft is not considered in this work. Sun illumination is also simulated by a spot light located around the scene (see Figure 1.11), setting up a quite decently realistic space context, with light/dark and specular effects on the target.
Tracking results can be observed on Figure 4.30 for $[C0, C1, C3, C4]$.

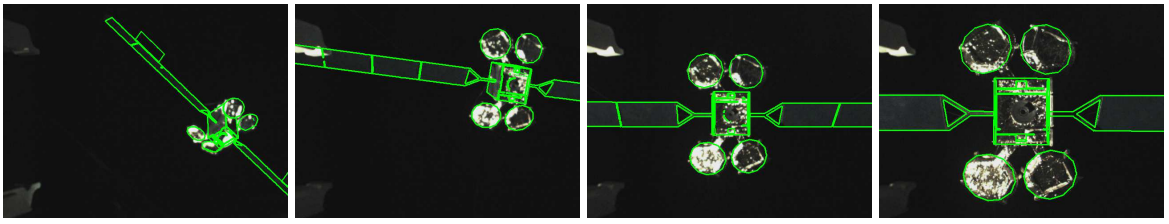
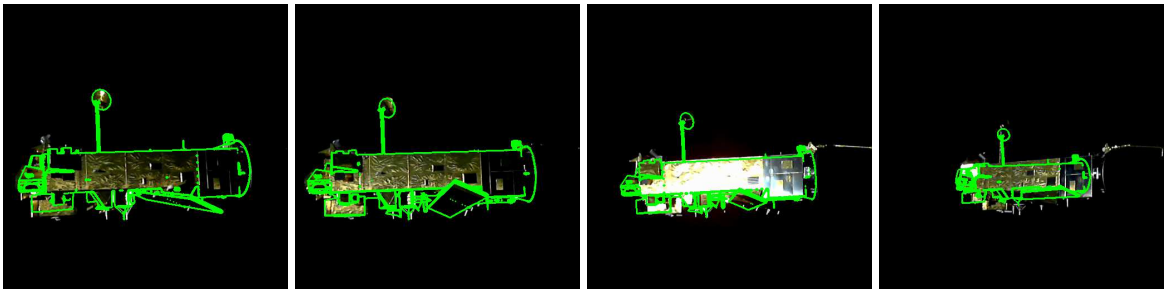


Figure 4.30 – Tracking results for the sequences involving Amazonas satellite mock-up.

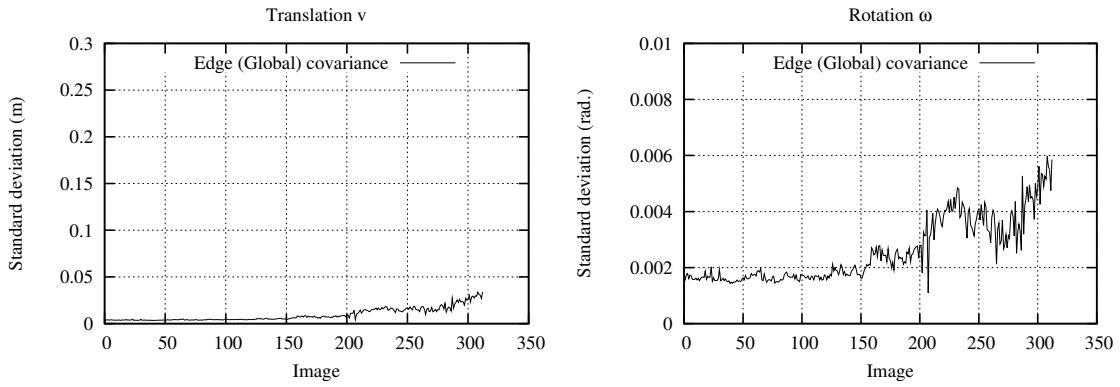
The Envisat sequence: the second sequences has been provided by Astrium and concerns a fly-around a mock-up of Envisat, an observation satellite which has been considered as a space debris since April 2012 and the sudden loss of contact with the satellite, for some inexplicable reason. Figures 4.31,4.32,4.33 respectively show tracking performed by $[C0, C1]$, $[C0, C1, C3, C4]$ and $[C0, C1, C3, C4, C5]$, along with the corresponding covariance parameters (Figures 4.31(e), 4.32(g), 4.33(g)). In order to show the uncertainty induced by each of the visual cue involved in the different tested solutions, marginal covariances $\Sigma^{g}_{\delta \mathbf{r}}$ (Edge covariance), $\Sigma^{c}_{\delta \mathbf{r}}$ (Color covariance) and $\Sigma^{p}_{\delta \mathbf{r}}$ (Point covariance) are also investigated. For each matrix, parameters $\sigma_{\boldsymbol{v}}$ (equation (4.91)) and $\sigma_{\boldsymbol{\omega}}$(equation (4.92)) are thus evaluated over the sequence.

Similarly to the Amazonas sequence, complicated illumination conditions, involving

darkness, specularities, can be noticed. Besides, the motion of the satellite is here mostly rotational, on the y-axis of the camera frame, and can be hardly observable on some phases. For the three tested solutions, let us first note that the covariances are principally impacted for the rotation parameters, since a rotational motion is involved. With $[C0, C1]$ (Figure 4.31), tracking on this challenging sequence can be properly achieved until frame 200, the rotation around y-axis then failing to be tracked, as observed on Figure 4.31(d) and on Figure 4.31(e), with growing covariances. With $[C0, C1, C3, C4]$ (Figure 4.32), tracking is correct until frame 700, as observed on Figure 4.32(f). The growth of the global covariance at this moment is noticeable, though slight. The larger uncertainty observed for the color-based features can be explained by the challenging color separation between the object and the background, due to the poor color contrast. The addition of $C5$ (interest points) allows slight improvements, tracking being correctly achieved until frame 760 (Figure 4.33(f). The marginal covariance on the tracked interest points shows a quite erratic behavior, with some peaks, especially between frame 400 and 500. This trend can be justified by the low luminosity (see frame 420 on Figure 4.33(d) for instance), which causes the extraction of few interest points to be tracked. If the Jacobian matrices of some of these points show low values, for instance a point close to the optical axis or with a large depth, then uncertainty greatly propagates for the whole set of points, and it results in a large $\mathbf{\Sigma}^{p}_{\delta \mathbf{r}}$

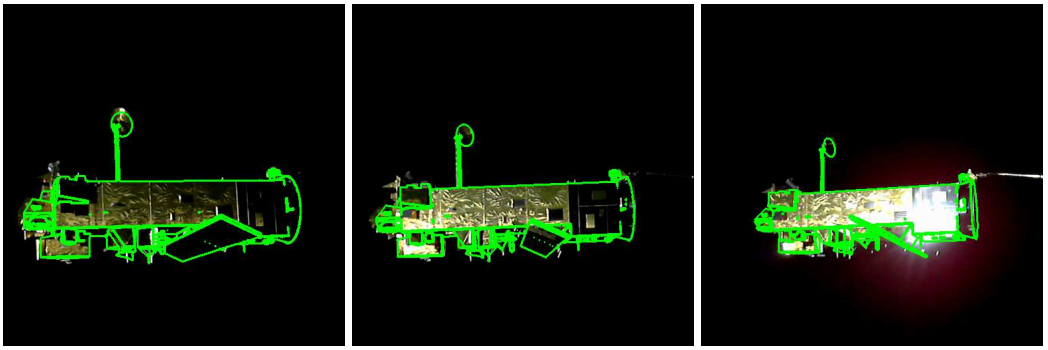


(a) frame 4 (b) frame 65 (c) frame 155 (d) frame 245

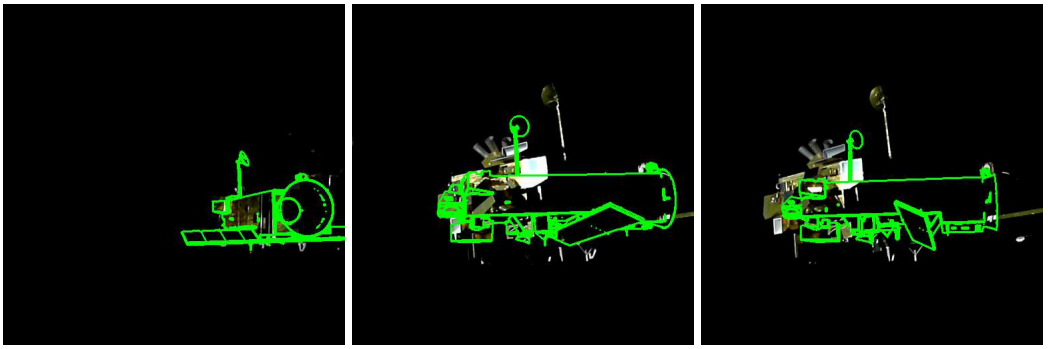


(e)

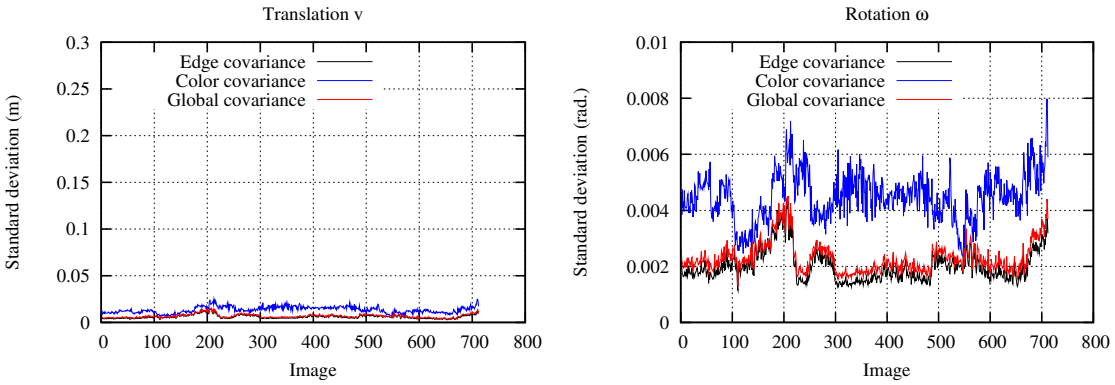
Figure 4.31 – Tracking for the Envisat sequence with $[C0, C1]$ and covariances on the camera pose error. $\mathbf{\Sigma}^{g}_{\delta \mathbf{r}}$ (Edge) is represented.



(a) frame 5 (b) frame 90 (c) frame 170

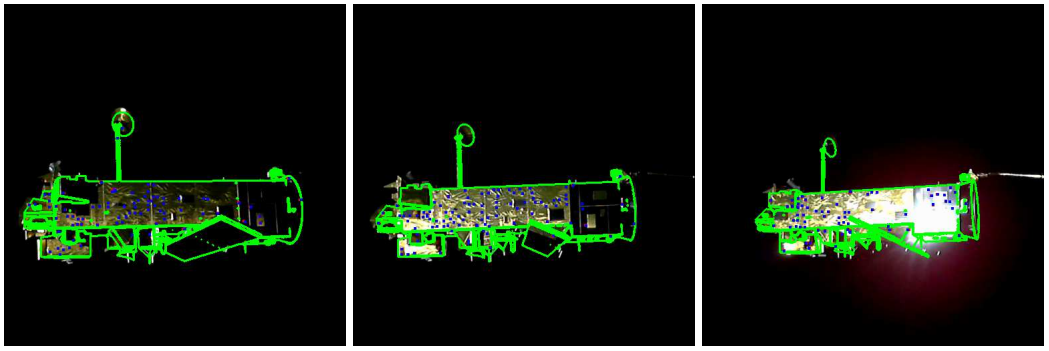


(d) frame 420 (e) frame 690 (f) frame 710

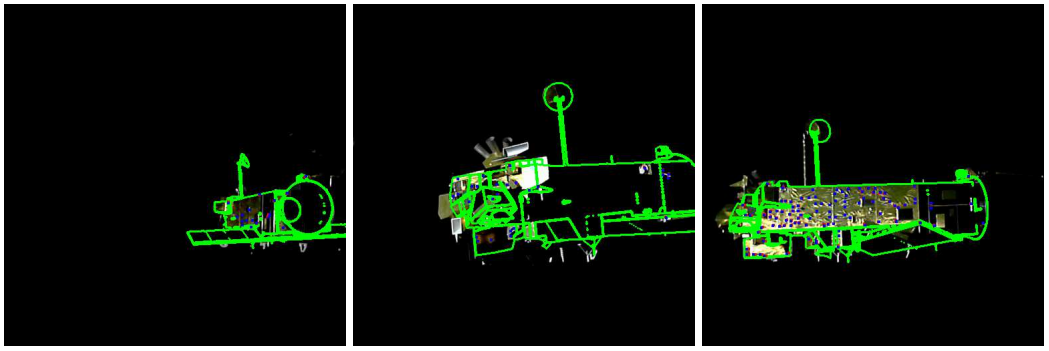


(g)

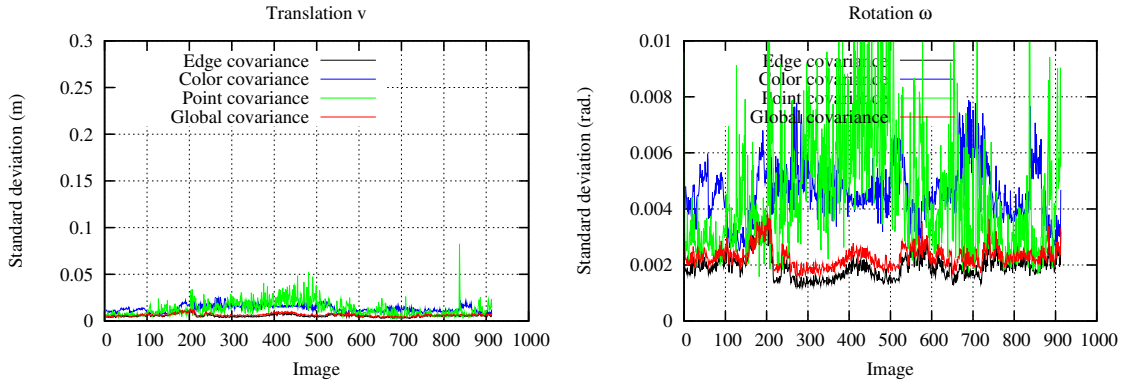
Figure 4.32 – Tracking for the Envisat sequence with $[C0, C1, C3, C4]$ and covariances on the camera pose error. $\mathbf{\Sigma}^{g}_{\delta\mathbf{r}}$ (Edge), $\mathbf{\Sigma}^{c}_{\delta\mathbf{r}}$ (Color) and $\mathbf{\Sigma}_{\delta\mathbf{r}}$ (Global) are represented.



(a) frame 5 (b) frame 90 (c) frame 170



(d) frame 420 (e) frame 690 (f) frame 760



(g)

Figure 4.33 – Tracking for the Envisat sequence with $[C0, C1, C3, C4, C5]$ and covariances on the camera pose error. $\boldsymbol{\Sigma}^{g}_{\delta \mathbf{r}}$ (Edge), $\boldsymbol{\Sigma}^{c}_{\delta \mathbf{r}}$ (Color), $\boldsymbol{\Sigma}^{p}_{\delta \mathbf{r}}$ (Point) and $\boldsymbol{\Sigma}_{\delta \mathbf{r}}$ (Global) are represented.

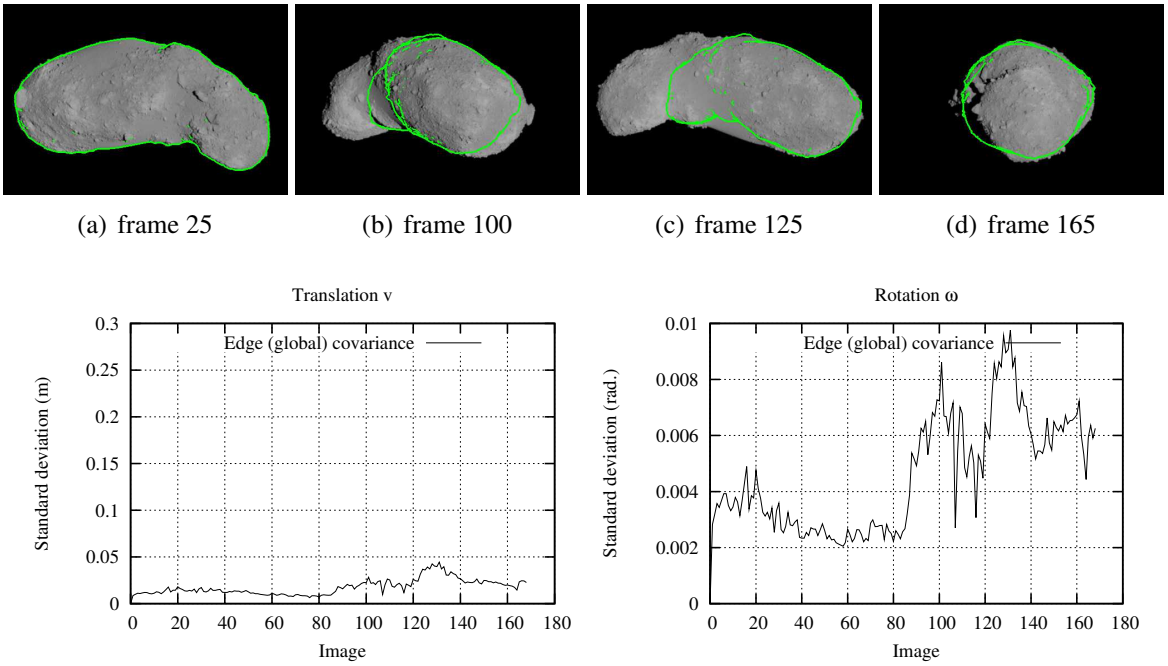
Itokawa asteroid sequence

Besides space rendezvous and proximity operation applications, we have also experimented our approaches to the case of navigation with respect to asteroids, for landing and sample return applications.

During the Hayabusa probe mission, a sequence of the Itokawa asteroid was captured, and a complete 3d model of the asteroid was reconstructed based on data collected by the probe and can be found on the Internet. We have directly processed the full model (30MB) using our efficient projection and edge generation system. The tested sequence, features a purely rotational motion of the target on the y-axis of the camera.

Solutions $[C0, C1]$, $[C0, C1, C3, C4]$ and $[C0, C1, C3, C4, C5]$ have been tested and respective tracking results are represented on Figures 4.34, 4.34, 4.35, 4.36, with their respective covariance parameters. With $[C0, C1]$, tracking fails around frame 90, as proven by Figure 4.34(b) for frame 100, and by the growth of the covariance from this frame (Figure 4.34(e)). Since a pure rotation is involved, mostly rotation parameters are impacted by the failure. Tracking also fails with $[C0, C1, C3, C4]$ around frame 80, due to some ambiguities on edges and on the shape, bringing the geometrical edge and color-based cues to local minima. For the covariance parameters (Figure 4.35(e)), a growth can be observed after frame 80 for the global covariance. This growth is however minor, especially with respect to the one observed for $[C0, C1]$. This can be explained by the fact that the uncertainty provided by the color-based features is quite small, since the color separation is very clear between the asteroid and the black background and since local minima are likely to be obtain due to ambiguities on the shape or silhouette. As an indicator, the uncertainty provided by interest points ($\mathbf{\Sigma}^p_{\delta \mathbf{r}}$), which are not taken into account in the global error function is represented, showing large values after the failure. The incorporation of interest points ($[C0, C1, C3, C4, C5]$) enables to track the asteroid over its whole rotational motion. As seen on Figure 4.36, numerous reliable Harris corners can be extracted and correctly tracked. The ability of these points to raise the ambiguity on silhouette edges can be stressed out. The covariances parameters (Figure 4.36(e)) can be kept low, with a standard deviation for the rotation parameters almost under 0.002 over the whole sequence. A peak can however be noticed for $\mathbf{\Sigma}^p_{\delta \mathbf{r}}$ around frame 80 (which is quite equivalent to frame 30 for the sequence presented on Figure 4.37). It can be justified by the fact that fewer interest points, mostly located around the image center, are extracted at this moment.

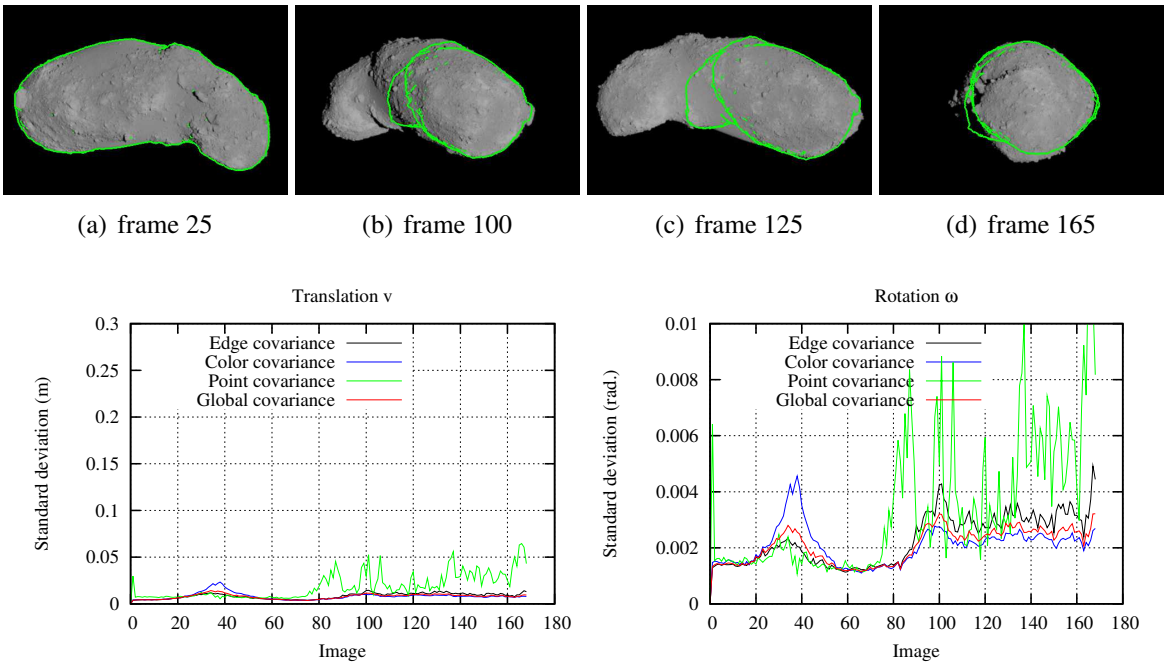
By down-sampling the sequence with $f = 3$, we are able to emphasize the benefit of out Kalman filtering and prediction frameworks. The resulting fast motion of the asteroid, with quite constant velocity, can be correctly tracked (Figure 4.37, bottom), based on the prediction step proposed on equation (4.90), in order to cope with the large motions. It fails with $[C0, C1, C3, C4, C5]$ (Figure 4.37, top).



(a) frame 25 (b) frame 100 (c) frame 125 (d) frame 165

(e)

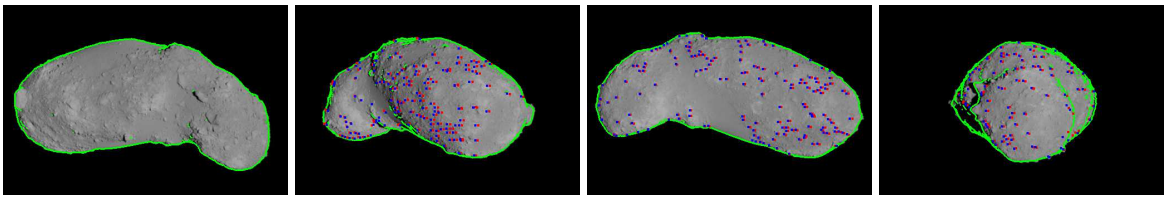
Figure 4.34 – Tracking for the Itokawa sequence with $[C0, C1]$ and covariances on the pose error. $\Sigma^{g}_{\delta \mathbf{r}}$ (Edge) is represented.



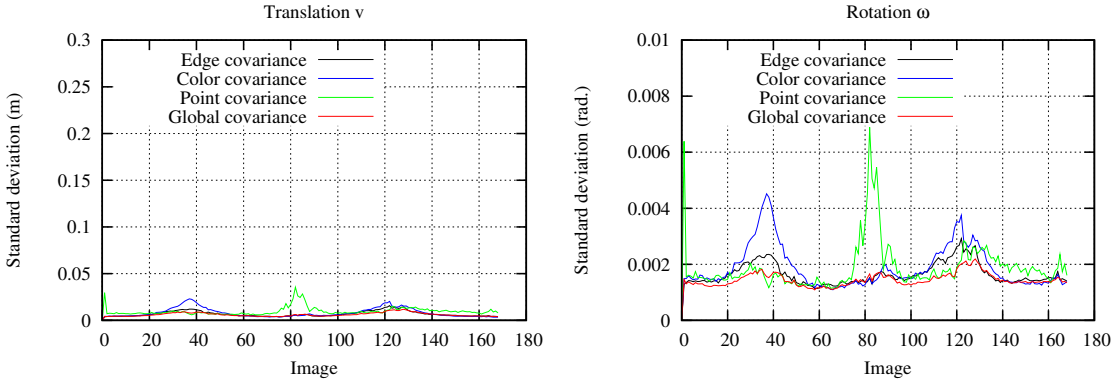
(a) frame 25 (b) frame 100 (c) frame 125 (d) frame 165

(e)

Figure 4.35 – Tracking for the Itokawa sequence with $[C0, C1, C3, C4]$ and covariances on the pose error. $\Sigma^{g}_{\delta \mathbf{r}}$ (Edge), $\Sigma^{c}_{\delta \mathbf{r}}$ (Color) and $\Sigma_{\delta \mathbf{r}}$ (Global) are represented. $\Sigma^{p}_{\delta \mathbf{r}}$ is represented but is not taken into account in the computation of $\Sigma_{\delta \mathbf{r}}$.

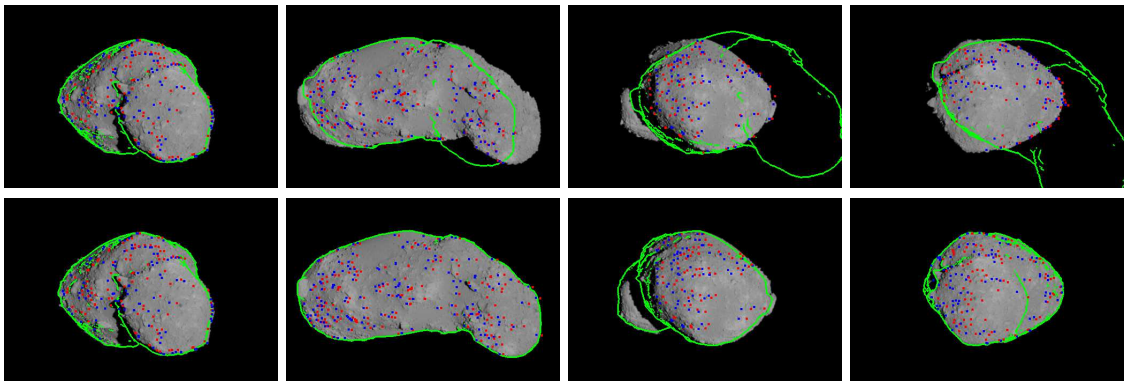


(a) frame 25 (b) frame 100 (c) frame 125 (d) frame 165



(e)

Figure 4.36 – Tracking for the Itokawa sequence with $[C0, C1, C3, C4, C5]$ and covariances on the pose error. $\Sigma^{g}_{\delta \mathbf{r}}$ (Edge), $\Sigma^{c}_{\delta \mathbf{r}}$ (Color), $\Sigma^{p}_{\delta \mathbf{r}}$ (Point) and $\Sigma_{\delta \mathbf{r}}$ (Global) are represented.



frame 1 frame 8 frame 30 frame 55

Figure 4.37 – Results with $[C0, C1, C3, C4, C5]$ (top) and $[C0, C1, C3, C4, C5, C6]$ (bottom) with $f = 3$.

4.6.1.4 Algorithm complexity and computational costs

In the suggested approaches, several issues are to be considered in the evaluation of the complexity of the algorithms.

- Size of the image, $H \times W$, with a H the height and W the width of the image.
- Size of the 3D model and number of generated model edge points N_g.
- Number of extracted and tracked Harris corners N_p.
- Number of iterations in the minimization process K.

For a given image size, we can evaluate and compare the overall complexity of the different solutions, only focusing the minimization process.

- With $C0$, the complexity C can be determined as: $C = O(N_g^2 K)$.
- The integration of multiple hypotheses $C1$, errors have to be computed for each hypothesis, in order to select the one with the smallest residue at each iteration of the minimization process. However the complexity of the computation of the errors can be neglected with respect to the computations of the Jacobian matrices, so that we also have: $C = O(N_g K)$.
- With $C2$ the complexity is also similar.
- With $C3$, an error function and a Jacobian matrix are computed for each point along the normal to the projected model edge points N_s belonging to the silhouette, and for each of the 3 color channel (RGB). Besides the computation of the local color statistics can be neglected with respect to the computations of errors and Jacobian matrices, giving: $C = O(6(L/\delta_D)N_s^2 K)$. By using luminance instead of colors, complexity can also be reduced to $C = O(2(L/\delta_D)N_s^2 K)$. The integration of $C4$ can be neglected.
- Using $C5$, the complexity is simply evaluated as: $C = O(N_p^2 K)$.

As a consequence, the complexity of the hybrid solution $[C0, C1, C2, C3, C4, C5]$, for instance, can be set, for the minimization process, to: $C = O((N_g + 6(L/\delta_D)N_s + N_p)^2 K)$.

Tables 4.2 and 4.3 respectively gather the number of processed control points and the different computational costs obtained for the different methods used to obtain results presented for the Mario sequence A.2 in appendix A, which is built of 1500 frames. In these cases, K is set to 10. D is set to 20 and H to 4, and 640×480 images are processed. Regarding N_g, it has been originally set to 300 and edge control points are then regularly sampled given this number, and the average resulting number of generated edge control points is given. Mean over the execution times on the considered sequence are presented, for the different phases of the pose estimation system, which are the projection and rendering of the 3D model, the generation of control points (egdes, silhouette edges, interest points), the low-level tracking (edge extraction and KLT algorithm) and the minimization process. For solutions including $C3$, the average number of generated silhouette edge control points N_s is also given, as well as the average number of extracted and tracked interest points when $C5$ is employed.

Data	$C0, C1$	$C0, C1, C3, C4$	$C0, C1, C3, C4, C5$
Number of edge points	310	307	306
Number of silhouette edge points		202	197
Number of interest points			32

Table 4.2 – Processed control points: number of edge points, silhouette edge points, and Harris corners interest points processed for the different methods

Phases	$C0, C1$	$C0, C1, C3, C4$	$C0, C1, C3, C4, C5$
Model projection and rendering (ms)	21.48	18.37	19.39
Generation of points (ms)	15.26	15.59	85.92
Low-level tracking (ms)	3.28	2.84	19.54
Minimization process (ms)	11.24	42.3	45.58
Total (ms)	51.26	79.1	170.43

Table 4.3 – Execution times.

We can remark that the rendering phase is quite heavy, around $20ms$, what is due to the size of the processed 3D model which 5.5MB, with 15000 vertices. Another observation to stress out is the weight of the generation of control points based on Harris corners extraction, which is quite costly, as well as the KLT algorithm used to track them. The extra complexity gendered by the addition of $C3$, and to a lesser extent of $C5$, since much fewer interest points than edge points are processed, can also be noticed for the minimization process. Let us finally point out that incorporating the Kalman filtering framework is negligeable in terms of computations.

4.6.2 Feasibility of space rendezvous using visual servoing

The idea of the following tests is to carry out a simulation of a closed loop rendezvous approach between the robotic arm available on the Lagadic robotic platform and the mock-up of the telecommunication satellite provided by Astrium, using visual servoing.

2 1/2 D visual servoing

In order to servo the robot, we propose to use the 3D-model based tracking algorithm within a 2 1/2 D visual servoing control loop. Visual servoing consists in using data provided by a vision sensor for controlling the motions of a dynamic system [Chaumette 06]. Classically, to achieve a visual servoing task, a set of visual features $\mathbf{s}$ has to be selected from the image to control the desired degrees of freedom. The goal is to minimize the error between the current values of visual features $\mathbf{s}$ extracted from the current image and their desired values $\mathbf{s}^*$. For this purpose, techniques [Chaumette 06] depend on the features $\mathbf{s}$ used : they can be 2D points directly extracted from the image, for Image-based Visual Servoing (IBVS) or 3D parameters recovered thanks to image measurements like pose computation for Position-based Visual Servoing (PBVS). Here we apply a hybrid solution, 2 1/2 D visual servoing approach [Chaumette 07, Chaumette 00], which avoids the shortcomings of the two basic approaches, by combining features in 2D and 3D, in order to decouple position and rotational movements, with a simpler interaction matrix,

and with a better stability than IBVS or PBVS :

$$\mathbf{s} = \begin{bmatrix} x & y & \theta u_z & \mathbf{t} \end{bmatrix}^T \tag{4.93}$$

where x and y are the metric coordinates in the image of a point of the object, here the center of the mock-up, θu_z is the third coordinate of the θu vector, which represents the rotation the camera has to perform to reach the desired pose, and $\mathbf{t}$ is the translation vector the camera has to perform to reach the desired pose, expressed in the desired camera frame. θu_z and $\mathbf{t}$ have thus to be regulated to 0. We need to minimize the error $\mathbf{e} = \mathbf{s} - \mathbf{s}^*$ where features $\mathbf{s}$ are recovered thanks to model-based tracking. A kinematic controller, which is convenient for most of robot arms, is then designed to servo the robot. A proportional control scheme is defined, to make the error exponentially decreases, leading to the following control law:

$$\mathbf{v_c} = -\lambda \widehat{\mathbf{L_s}}^{+} (\mathbf{s} - \mathbf{s}^*) \tag{4.94}$$

with $\widehat{\mathbf{L_s}}^{+}$ the estimate of the pseudo-inverse of $\mathbf{L_s}$, the interaction matrix associated to the visual features. The paper described in [Chaumette 06] details how this matrix can be computed. It can here be estimated thanks to the parameters of the pose computed by the model-based tracking algorithm.

Experimental set-up and results

The experimental set up is very similar to the one depicted on Figure 1.11. Regarding the simulated rendezvous maneuver, it has been divided into two phases: a first displacement to bring the target into the center of the image and to realign to the docking port axis of the target, and a final translation until a secure distance to the mock-up docking. Three different illumination conditions have been tried out: favorable, almost darkness and variable with harsh changes. The servoing performed on the mock-up has successfully achieved the intended goal, for the three different illumination conditions (Figure 4.38, 4.39, 4.40). The model-based solution adopted for these tests is $[C0, C1, C3, C4, C5]$. Figure 4.41 presents results obtained with a proportional gain λ twice the one used for the former experiments, with $[C0, C1, C3, C4, C5]$. Let us note that with such a gain tracking solution $[C0, C1, C3, C4]$ is unable to perform correctly due to the larger inter-frame, making the servoing task fail.

4.7 Conclusion

In this chapter we have presented our solution to address the problem of efficiently and robustly estimating the pose between a camera and a target object in an image sequence by tracking it frame-to-frame, based on the 3D model of the target. With respect to state-of-the-art approaches, the overall goal of our approach has been to provide improvements on several aspects of the problem:

- **Implementing an efficient system to project the 3D model and generate a set of control points from the projection**. Through this phase, the aim is to be able to deal with complete 3D models of any kind of object. This task has been handled by using a 3D rendering engine and GPU-based operations to generate the control points.

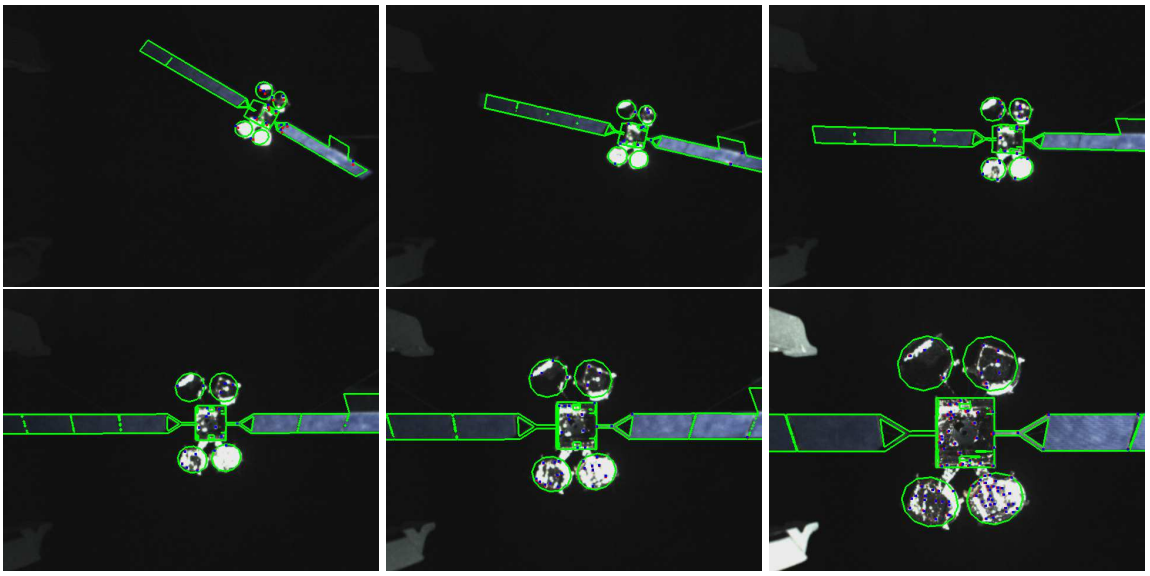


Figure 4.38 – 2 1/2D visual servoing under favorable illumination conditions for [$C0, C1, C3, C4, C5$].

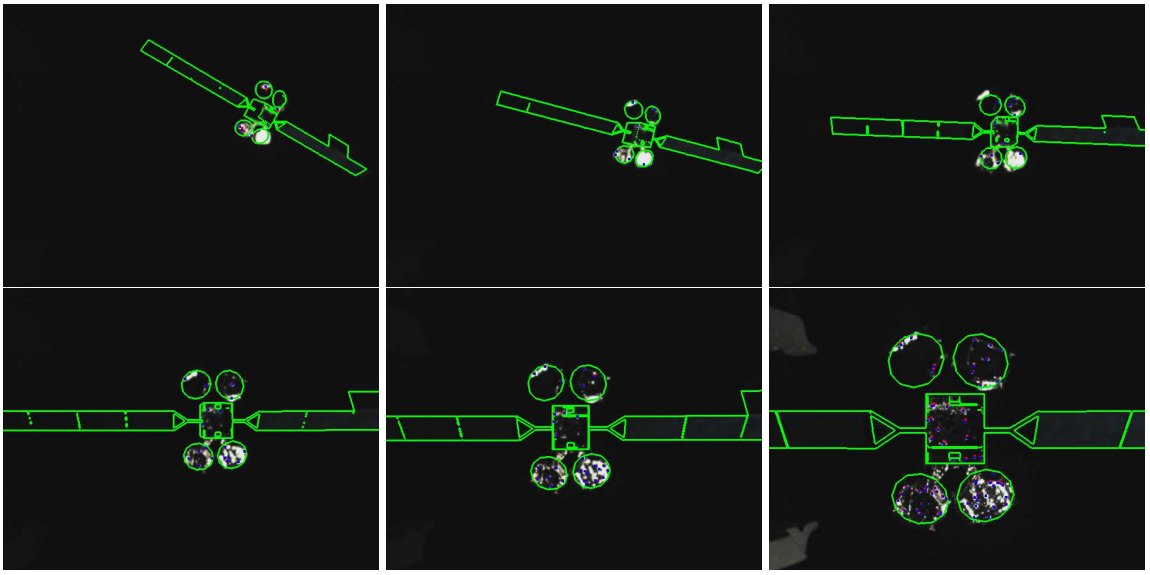


Figure 4.39 – 2 1/2D visual servoing under weak illumination conditions for [$C0, C1, C3, C4, C5$].

- **Designing a robust pose estimation process**. Based on the projection of the 3D model or the generated 3D control points, pose estimation relies on a deterministic non-linear minimization process of an error function accounting for some visual information extracted from the image. With the general idea that a 3D object can be quite fully and pertinently representedby its edges, its shape or silhouette and by a set of interest points such as corners, our contributions have consisted in combining in the global criterion to be minimized three different visual cues related to these three different representation modes. For this purpose we have elaborated two geometrical cues, relying on distances between edge features and interest point features, and a intensity-based one, relying on color features computed along the

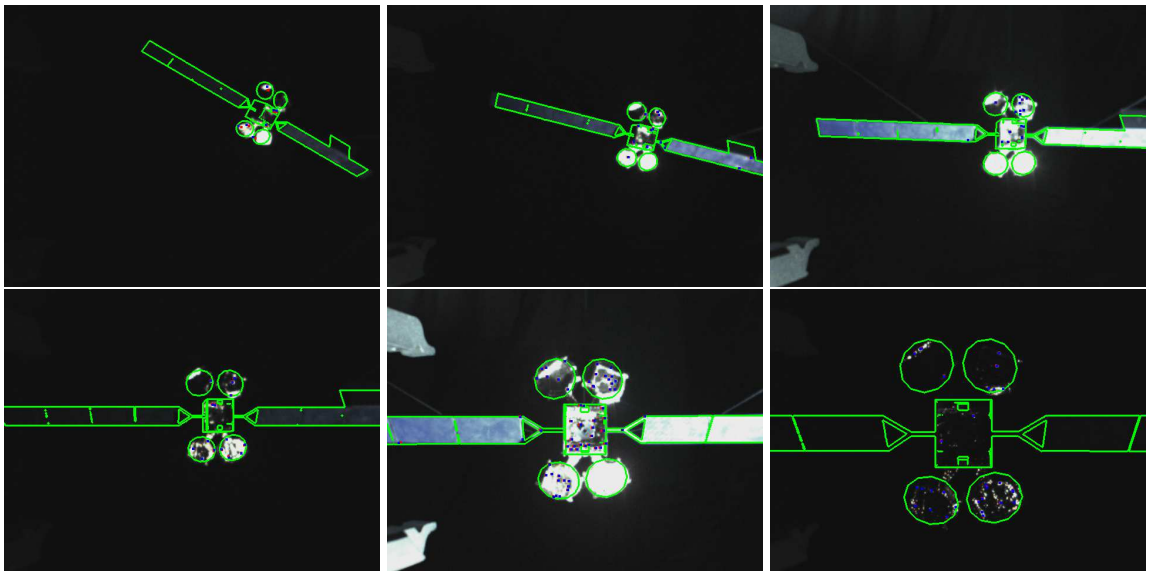


Figure 4.40 – 2 1/2D visual servoing under variable illumination conditions and with a higher gain, for [$C0, C1, C3, C4, C5$].

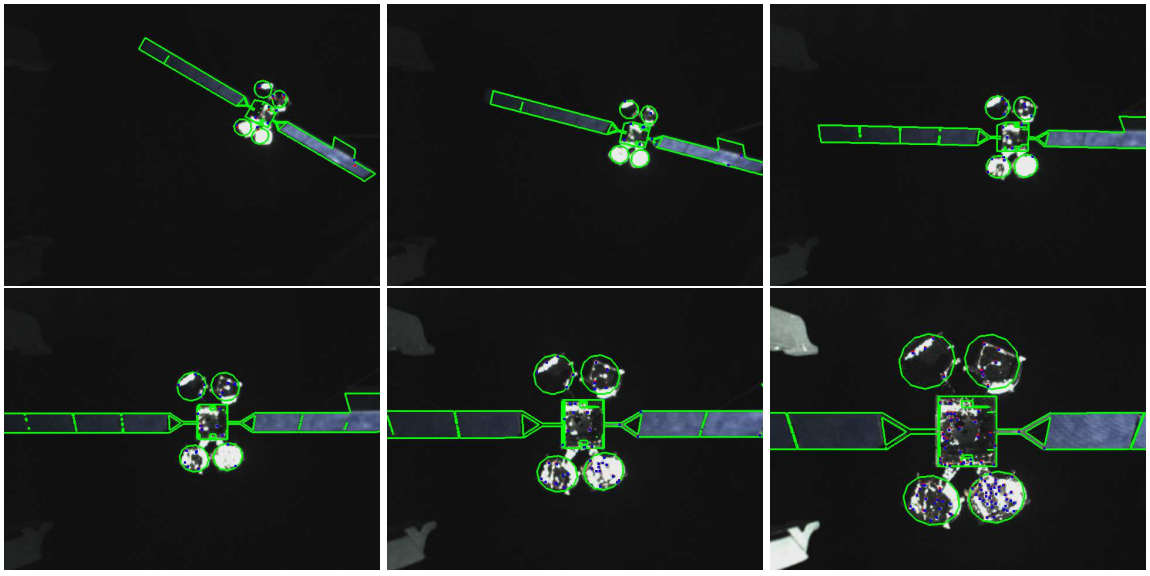


Figure 4.41 – 2 1/2D visual servoing under variable illumination conditions for [$C0, C1, C3, C4, C5$].

silhouette edges, each cue providing complementary benefits.

- **Characterizing the reliability of the tracking process**. By using uncertainty propagation, a global and marginal (for each visual cue) covariance matrices on the pose errors parameters can be obtained, from the low-level uncertainty of the visual features. Based on this uncertainty, a Kalman filtering and pose prediction module has been designed, to smooth pose estimates and to cope with large inter-frame displacements.

Our approach has been tested via numerous experiments, on both synthetic and real images. The objective has been to validate our contributions under various conditions.

The tried out sequences indeed feature various complex 3D objects, various motions of these targets, various illumination conditions, with motion blur, with background clutter, noise... By dealing with synthetic images, quantitative and comparative evaluations of our solutions have been carried out. All these tests have shown the benefits of each visual cues: robustness with respect to illumination conditions for the geometrical edge-based criterion, accuracy and robustness to motion blur for color-based features, temporal smoothness and robustness to large motions for interest points. Finally, the Kalman filtering and prediction framework is able to smooth pose estimates and to deal with large motions through the prediction step. Through our efficient projection system of the 3D model, any sort of 3D models can be processed, with reasonable execution times. Most of the experiments have dealt with space objects for space rendezvous navigation applications. The benefit of our method has made it suitable for visual servoing purposes, showing the feasibility of a pure vision-based close range rendezvous mission. However, the generic nature of our tracking algorithms makes them also applicable for other fields, such as augmented reality, on any kind of complex objects.

Conclusion and perspectives

The challenge of autonomy for robotic systems involves the need for a reliable sensing technology. When dealing with localization issues, cameras are a common and preferred choice in many applications. Through the design of pattern recognition, motion analysis or tracking algorithms, computer vision can supply the considered robotic system with accurate and robust visual localization capability with respect to a known or unknown environment.

In space robotics, for technical, economic or safety reasons, the topical and critical problems of on-orbit servicing and space debris removal has involved an accrued interest on incorporating a high-level of autonomy for the related space rendezvous and proximity operations. In order to handle uncooperative targets, the use of monocular cameras and computer vision appear as a pertinent choice to accurately localize the target, given the stringent navigation measurement requirements involved for such an operation.

In this thesis we have proposed a unified solution to address the task of fully localizing a known object with respect to a monocular camera, with a focus on the navigation concerns of space autonomous rendezvous operations. We suppose the knowledge of the 3D model of the object. This unified solution first consists in visually detecting and initially fully localizing the target object. Then, the output of this detection module initializes a visual localization system of the target using frame-by-frame tracking.

The issue of the **detection and initial localization** step is addressed through pose estimation based on a set of initial images. The approach we propose lies in the field of template matching methods, where the templates are synthetic views of the object generated using its a priori known 3D CAD model.

A first step consists in classifying the views into a hierarchical view graph to efficiently handle the large search space. Assuming the single foreground object is moving in the image with respect to the background, our method then benefits from a foreground/background segmentation to guide a registration procedure of the reference views with the set of initial input image. From the most likely view, a pose refinement is finally performed via a best match search in the view graph.

Among our contributions, we can mention our foreground/background segmentation technique which is particularly suited for the context of space rendezvous. We have also elaborated a robust edge-based similarity measure, for both the learning and the online matching process, whose advantage with respect to a classical measure has been demonstrated. A last notable contribution is the incorporation of particle filtering to refine similarity transformation parameters.

The results, performed on various objects (more or less textured) and imaging conditions (synthetic images, real images, cluttered background, specular effects) have shown the efficiency and the robustness of the method to these challenging conditions, as well as its ability to provide a sufficiently precise pose to initialize the tracking step.

For the **visual tracking step**, it is based on pose estimation through a 3D model-based tracking algorithm. In this work, we relied on a classical deterministic non-linear optimization framework, intended to minimize an error between visual features observed in the image and the forward projection of their 3D homologues, using the 3D model. We propose three different types of visual features which enable to pertinently represent the object with its whole set of edges, its silhouette and with a set of interest points, providing two geometrical errors (edges and interest points) and an intensity-based error (colors along silhouette edges). The main contributions of this solution has been to implement an efficient projection system of a potentially complete and large 3D model, to handle any sort of complex objects, and to combine these complementary visual cues. Other contributions principally involve the designs of multiple-hypotheses frameworks for the geometrical edge-based registration process and the addition of a temporal constraint on the color features.

In order to determine an indicator of the **uncertainty of the localization process**, we suggest to propagate the uncertainty from the low-level errors of the visual features, giving a covariance matrix on the pose error. This uncertainty feeds a linear Kalman filter on the camera velocity parameters. We propose to use this Kalman filtering framework for both smoothing pose estimates and predicting the pose for the next frame, enabling to handle large displacements.

Comparative experiments on various objects and with various conditions (illumination, specular effects, background...) have been carried out, showing the advantages of the different contributions, quantitatively and qualitatively. Besides its relevance regarding space rendezvous issues with a space object such as a satellite, this method is also suitable for any other complex scenes (asteroid navigation, augmented reality).

Discussions and perspectives

Detection and initial pose estimation

Our detection and initial pose estimation has proven its efficiency when the moving target can be segmented from the background, especially in the case of a space context with "deep space" black background or "terrestrial" backgrounds. Although it can be robust to some segmentation errors, thanks to the alignment procedure, it can still be sensitive to this step. For both terrestrial backgrounds, a solution would be to use the localization information provided by chaser spacecraft with respect to the earth to provide prior information on the earth apparent motion. Otherwise, some priors on the apparent motion or color of the earth surface could be used, as in [Criminisi 06, Yin 07], requiring learning steps but relaxing our assumptions quite specific to our application (stationary and planar background). Without learning or supervision, the approach proposed in [Brox 10], in a similar spirit to ours regarding clustering of foreground/background trajectories, would be interesting.

Besides, in order to make our matching and alignment process of the views more

generic, we could relax the prior step of segmenting the moving target from the background. For this purpose, we could think of integrating some region or part descriptors based learning methods such as the promising one proposed in [Stark 10]. The idea would be to learn local salient parts of the 3D model into a probabilistic spatial model with respect to the pose parameters, using visual descriptors such as *Shape Context* [Belongie 02] or HOG [Dalal 05]. This probabilistic model would be inferred by processing correspondences between the learned descriptors and descriptors provided by the image, providing the pose or a viewpoint estimate. In [Stark 10] viewpoints or pose estimates still appear to be coarse. By combining this method with our template matching and alignment procedure, an accurate pose could be obtained. The method in [Payet 11], which relies on an edge-based template matching strategy, using natural training images, could be interesting to experiment, by applying it to a set of non-photorealistic synthetic views of the object.

Tracking

Regarding visual tracking issues, our solution considers a known object, and relies on a prior fixed accurate 3D CAD model of the object to perform correctly. In the case of a partially known object, with missing parts (such as damaged space debris), or with a partial or coarse 3D model, our tracking and pose estimation algorithm would be enhanced and ameliorated by Simultaneaous Localization and Mapping (SLAM) techniques. As suggested in [Tamaazousti 11, Prisacariu 13b, Prisacariu 13a], a partial [Tamaazousti 11] or a very coarse [Prisacariu 13b, Prisacariu 13a] 3D model could be processed in a SLAM system, using points [Tamaazousti 11] or shape information [Prisacariu 13b, Prisacariu 13a], to simultaneously reconstruct the whole scene or refine the 3D model, pose estimation benefiting from this reconstruction.

Let us also note that our tracking approach could easily be implemented to deal with multiple objects, or, with few improvements, with articulated objects, in the case of the solar panels of a satellite for instance.

Space robotics applications and others

With the works proposed in this document, we have essentially focused on space rendezvous and proximity operations applications.

For potential on-orbit implementation of our solution, two issues shall be discussed:

- First, our visual tracking localization system, based on a model-based tracking algorithm, relies on hardware acceleration using the Graphics Process Units (GPU). The projection of the 3D CAD model and the generation of the control points (for the edges, the silhouette, and the interest points) is managed through a 3D rendering engine. This is essential to automatically and efficiently handle potential complete 3D models of complex objects, achieving reasonable computational performances, independently of the complexity of the 3D model.

  However, GPUs are not likely to be embedded onto spacecrafts in a very near future, but the parallel architecture of actual on-board processors could be sufficient to handle the problem. Another solution would be to deal with lighter pre-processed 3D models, and to rely on the implementations of the model projection system proposed in [Comport 06b, Petit 11].

- This document provides full localization solutions for navigation purposes in the context of the final approach of a space rendezvous. As shown by the feasibility study of a closed Guidance Navigation Control (GNC) loop, using visual servoing on a mock-up of a telecommunication satellite (section 4.6.2), under challenging imaging conditions, the detection and tracking solutions could be directly integrated in such a loop. However, this study does not consider the specific dynamics of the chaser spacecraft (or approximates it by a simple integrator). Further works could aim at investigating realistic control issues using this sensing module, as it has been performed, in simulation, for aircraft landing applications for instance [Coutard 11b].

In this study, we have performed experiments on some very challenging image sequences, involving difficult illumination conditions, large motions, noise, to comparatively emphasize the advantages of the different contributions, their robustness and their applicability for space robotics concerns. Let us note that failing solutions on these sequences ([$C0$],[$C3$] for instance) shall work properly on more conventional data, for space applications or others.

Finally, other applications of our unified solution (both initial pose estimation and tracking) can be considered. We can think of aerospace applications such as aircraft refueling, aircraft landing, for instance on carriers [Coutard 11a, Coutard 11b] or planetary landing of spacecrafts or probes, as evoked by our tests on the Itokawa asteroid. But we can also think of completely different application fields such as augmented reality or any robotic systems involving a monocular camera and a known 3D object.

Appendix A

Augmented Reality applications

Mario sequence and augmented reality applications

Our tracking method has been tested on a figurine of "Mario". The target, which is 35cm high, is made of curved and complex shapes. Since no 3D model is available, we have reconstructed it automatically using a Kinect sensor and the ReconstructMe software, which is an easy to use real-time 3D reconstruction system. Views of the resulting 3D model can be seen on Figure A.1. Despite rough modelization of some parts (on the cap, the hands for instance), this model, which is made of 15000 vertices, for a 5.5MB size, has been, as for the Itokawa asteroid, directly used and processed in our tracking algorithm, showing the convenience of the proposed method.

Figure A.1 – Views of the reconstructed 3D model of Mario.

The results of some solutions are shown on Figures A.2 and A.3. $[C0, C1]$, $[C0, C1, C3, C4]$, $[C0, C1, C3, C4, C5, C6]$, are able to correctly track the object on the whole sequence, whereas $[C0]$, $[C3]$, $[C0, C3]$ fail. The incorporation of multiple hypothesis for edges, of the color features, of the interest point features and of the Kalman filter, whose prediction scheme is based on equation 4.89, provides less sensitivity to local minima, more accuracy, temporal constraint, smoothness and the ability to handle large motions.

With $[C0, C1, C3, C4]$, tracking gets quickly lost (Figure A.4(b)) due to the large inter-frame motion, whereas $[C0, C1, C3, C4, C5]$ achieves it successfully throughout the sequence, even in the case of very large motions in the image (up to 40 pixels) and important motion blur (see Figure A.5(a)).

The uncertainty of the pose for both methods is also represented (Figures A.4(c), A.5(e)), for the different visual features, with $\mathbf{\Sigma}^{g}_{\delta \mathbf{r}}$, $\mathbf{\Sigma}^{c}_{\delta \mathbf{r}}$, and $\mathbf{\Sigma}^{p}_{\delta \mathbf{r}}$, and for the whole set with $\mathbf{\Sigma}_{\delta \mathbf{r}}$. On Figure A.4(c) $\mathbf{\Sigma}^{p}_{\delta \mathbf{r}}$ is represented but is not taken into account in the computation of $\mathbf{\Sigma}_{\delta \mathbf{r}}$. We can observe that the effective fail of the tracking for $[C0, C1, C3, C4, C5]$ around frame 5 does not have much effect on the global covariance, however, the covariances generated by the interest points and the geometrical edge features take much larger values.

We have augmented this object with a virtual "Yoshi", whose 3D model was found on Google 3DWarehouse. The rendering process of the 3D models of both the tracked and the augmenting objects also relies on OpenSceneGraph. Figures A.6 shows the results of the tracking and the corresponding augmenting task on two different sequences, with fixed and hand held cameras (see provided videos). Despite the motion blur, cluttered background or shaky displacements the object is correctly augmented throughout the sequences.

frame 8 frame 25 frame 65 frame 125

Figure A.2 – Results, from top to bottom with [$C0$], [$C3$] and [$C0, C3$].

frame 8 frame 25 frame 135 frame 915

Figure A.3 – Results, from top to bottom with [$C0, C1$], [$C0, C1, C3, C4$] and [$C0, C1, C3, C4, C5, C6$].

(a) frame 12 (b) frame 15

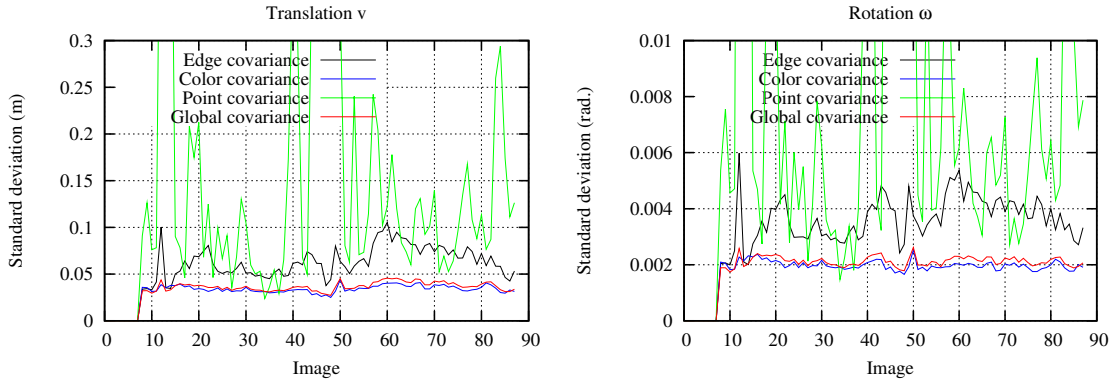
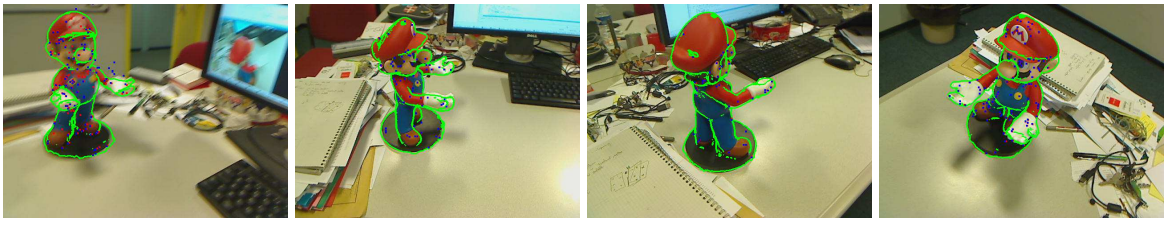


Figure A.4 – Tracking for the Mario sequence with $[C0, C1, C3, C4]$ and covariances on the pose error. $\Sigma^{g}_{\delta \mathbf{r}}$ (Edge), $\Sigma^{c}_{\delta \mathbf{r}}$ (Color), $\Sigma_{\delta \mathbf{r}}$ (Global) are represented. $\Sigma^{p}_{\delta \mathbf{r}}$ is represented but is not taken into account in the computation of $\Sigma_{\delta \mathbf{r}}$.



(a) frame 15 (b) frame 205 (c) frame 260 (d) frame 495

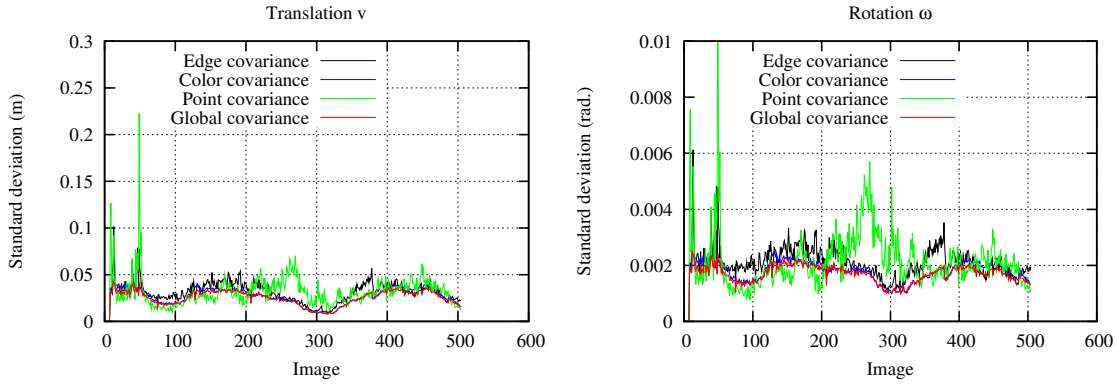


Figure A.5 – Tracking for the Mario sequence with $[C0, C1, C3, C4, C5]$ and covariances on the pose error. Square root of traces on translation and rotation parameters of $\Sigma^{g}_{\delta \mathbf{r}}$ (Edge), $\Sigma^{c}_{\delta \mathbf{r}}$ (Color), $\Sigma^{p}_{\delta \mathbf{r}}$ (Point) and $\Sigma_{\delta \mathbf{r}}$ (Global) are represented.

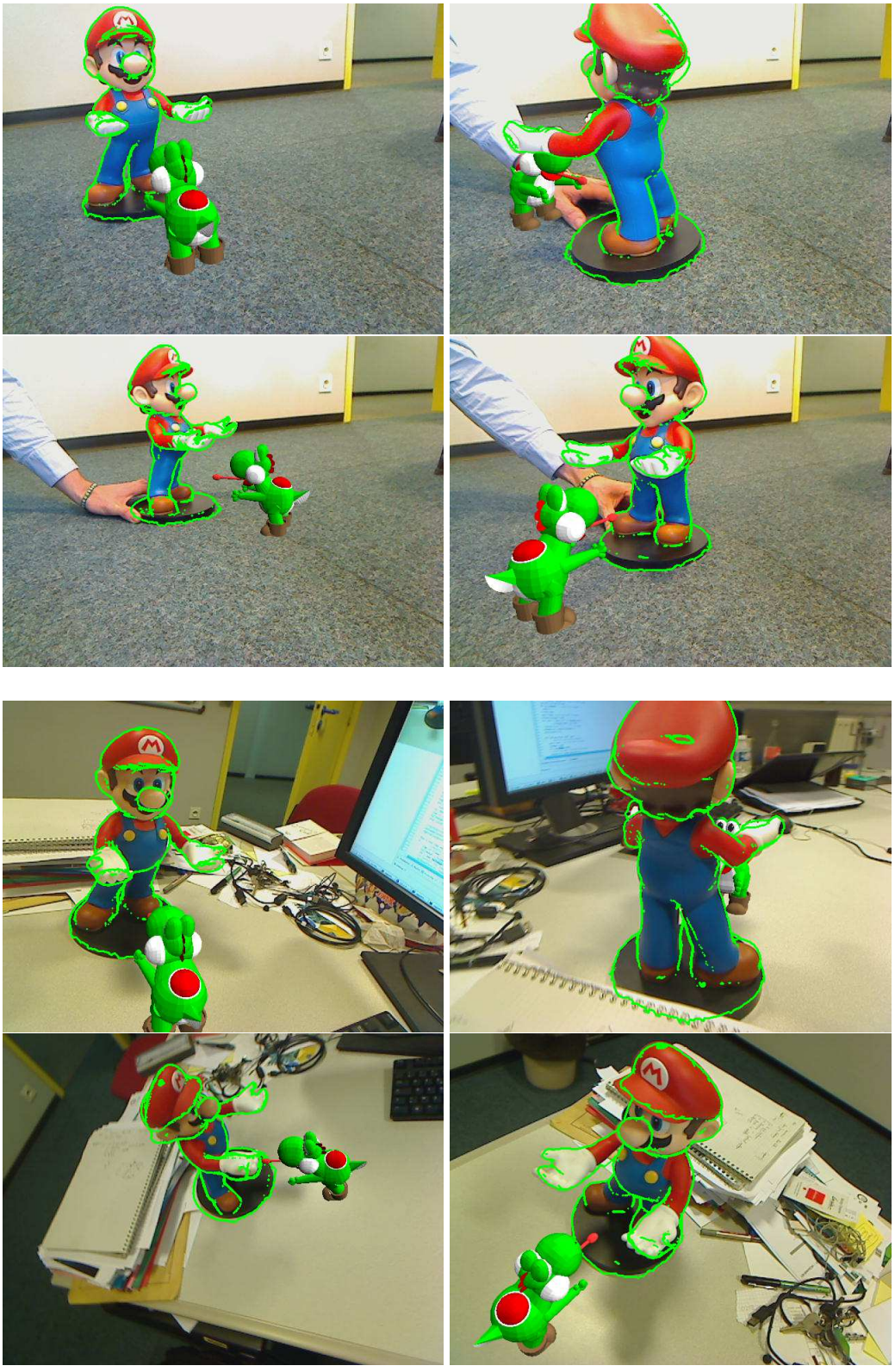


Figure A.6 – Tracking and augmenting Mario.

Bibliography

[Aach 95] T. Aach, A. Kaup. – Bayesian algorithms for adaptive change detection in image sequences using markov random fields. *Signal Processing: Image Communication*, 7(2):147–160, 1995.

[Abramowitz 64] M. Abramowitz, I.A. Stegun. – *Handbook of mathematical functions with formulas, graphs, and mathematical tables*. – U.S. Government Printing Office, vol. 55 of *National Bureau of Standards Applied Mathematics Series*, xiv+1046p., Washington, D.C., 1964.

[Amit 04] Y. Amit, D. Geman, X. Fan. – A coarse-to-fine strategy for multiclass shape detection. *IEEE Trans. on PAMI*, 26(12):1606–1621, décembre 2004.

[Armstrong 95] M. Armstrong, A. Zisserman. – Robust object tracking. 1995.

[Astrium 10] EADS Astrium. – Integrated mutli-range rendez-vous control system and autnomous rendez-vous and capture gnc test facility - summary report. 2010.

[Augenstein 11] S. Augenstein. – *Monocular Pose and Shape Estimation of Moving Targets for Autonomous Rendezvous and Docking*. – Stanford University, 2011.

[Barrow 77] H. Barrow, J. Tenenbaum, R. Bolles, H. Wolf. – *Parametric correspondence and chamfer matching: Two new techniques for image matching*. – Rapport de recherche, DTIC Document, 1977.

[BarShalom 93] Y. Bar-Shalom, X.-R. Li. – *Estimation and Tracking, Principles, Techniques, and Software.* – Artech House, Boston, 1993.

[Bascle 94] B. Bascle, P. Bouthemy, N. Deriche, F. Meyer. – Tracking complex primitives in an image sequence. – *Int. Conf. on Pattern Recognition, ICPR'94*, pp. 426–431, Jerusalem, octobre 1994.

[Basu 96] S. Basu, I. Essa, A. Pentland. – Motion regularization for model-based head tracking. – *IAPR Int. Conf. on Pattern Recognition*, vol. 3, pp. 611–616, Vienna, Austria, 1996. IEEE.

[Baumberg 00] A. Baumberg. – Reliable feature matching across widely separated views. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 1, pp. 774–781, Hilton Head, SC, 2000. IEEE.

[Bay 06] H. Bay, T. Tuytelaars, L. Van Gool. – Surf: Speeded up robust features. – *European Conf. on Computer Vision*, pp. 404–417, 2006.

[Beaton 74] A.E. Beaton, J.W. Tukey. – The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, 16:147–185, 1974.

[Belongie 02] S. Belongie, J. Malik, J. Puzicha. – Shape matching and object recognition using shape contexts. *IEEE Trans. on PAMI*, 24(4):509–522, April 2002.

[Benhimane 04] S. Benhimane, E. Malis. – Real-time image-based tracking of planes using efficient second-order minimization. – *IEEE/RSJ Int. Conf. on Intelligent Robots Systems*, vol. 943-948, p. 1, Sendai, Japan, octobre 2004.

[Blais 10] F. Blais, J.A. Beraldin, L. Cournoyer, I. Christie, R. Serafini, K. Mason, S. McCarthy, C. Goodall. – Integration of a tracking laser range camera with the photogrammetry based space vision system. – *SPIE Aerosense*, vol. 4025, p. 219, avril 2010.

[Blarre 04] L. Blarre, N. Perrimon, C. Moussu, P. Da Cunha, S. Strandmoe. – Atv videometer qualification. – *Proc. of the 55th Int. Astronautical Congress*, Vancouver, Canada, 2004.

[Bleser 05] G. Bleser, Y. Pastarmov, D. Stricker. – Real-time 3d camera tracking for industrial augmented reality applications. *Journal of WSCG*, pp. 47–54, 2005.

[Bonnal 13a] C. Bonnal. – Active debris removal: Current status of activities in cnes. *IAF Workshop on Space Debris Removal*, 2013.

[Bonnal 13b] C. Bonnal, J.M. Ruault, M.C. Desjean. – Active debris removal: Recent progress and current trends. *Acta Astronautica*, 85:51–60, 2013.

[Borgefors 88] G. Borgefors. – Hierarchical chamfer matching: A parametric edge matching algorithm. *IEEE Trans. on PAMI*, 10(6):849–865, juin 1988.

[Bouthemy 89] P. Bouthemy. – A maximum likelihood framework for determining moving edges. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(5):499–511, mai 1989.

[Boykov 01] Y. Boykov, O. Veksler, R. Zabih. – Fast approximate energy minimization via graph cuts. – *IEEE Trans. on PAMI*, vol. 23, pp. 1222–1239, novembre 2001.

[Brown 71] D.C. Brown. – Close-range camera calibration. *Photogrammetric Engineering*, 4(2):127–140, mars 1971.

[Brown 92] C. Brown. – Issues in selective perception. – *IAPR Int. Conf. on Pattern Recognition, ICPR'92*, vol. 1, pp. 21–24, The Hague, Netherland, août 1992.

[Brox 06] T. Brox, B. Rosenhahn, D. Cremers, H.-P. Seidel. – High accuracy optical flow serves 3-D pose tracking: exploiting contour and flow based constraints. – A. Leonardis, H. Bischof, A. Pinz (édité par), *European Conf. on Computer Vision, ECCV'06*, vol. 3952 of *LNCS*, pp. 98–111, Graz, Austria, May 2006. Springer.

[Brox 10] T. Brox, J. Malik. – Object segmentation by long term analysis of point trajectories. *Computer Vision–ECCV 2010*, pp. 282–295. – Springer, 2010.

[Brudieu 12] P Brudieu, B Lazare. – French policy for space sustainability and perspectives. – *16th ISU Symposium on Space Activities, Strasbourg*, 2012.

[Bugeau 07] A. Bugeau, P. Pérez. – Detection and segmentation of moving objects in highly dynamic scenes. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, Minneapolis, Minnesota, 2007. IEEE.

[Canny 86] J.F. Canny. – A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine intelligence*, 8(6):679–698, novembre 1986.

[Cavallaro 00] A. Cavallaro, T. Ebrahimi. – Video object extraction based on adaptive background and statistical change detection. – *Photonics West 2001-Electronic Imaging*, pp. 465–475. International Society for Optics and Photonics, 2000.

[Center 04] Marchall Space Flight Center. – Dart demonstrator to test future autonomous rendezvous technologies in orbit. – FS-2004-08-113-MSFC, septembre 2004.

[Chaumette 00] F. Chaumette, E. Malis. – 2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings. – *IEEE Int. Conf. on Robotics and Automation*, vol. 1, pp. 630–635, San Francisco, CA, avril 2000.

[Chaumette 06] F. Chaumette, S. Hutchinson. – Visual servo control, Part I: Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, December 2006.

[Chaumette 07] F. Chaumette, S. Hutchinson. – Visual servo control, Part II: Advanced approaches. *IEEE Robotics and Automation Magazine*, 14(1):109–118, March 2007.

[Choi 12] C. Choi, H. I. Christensen. – Robust 3d visual tracking using particle filtering on the special euclidean group: A combined approach of keypoint and edge features. *Int. Journal of Robotics Research*, 31(4):498–519, avril 2012.

[Christian 13] J. A. Christian, S. Cryan. – A survey of lidar technology and its use in spacecraft relative navigation. 2013.

[Cobzas 05] D. Cobzas, P. Sturm. – 3D SSD tracking with estimated 3D planes. – *The 2nd Canadian Conf. on Computer and Robot Vision*, pp. 129–134. IEEE, 2005.

[Collet 09] A. Collet, D. Berenson, S. Srinivasa, D. Ferguson. – Object recognition and full pose registration from a single image for robotic manipulation. – *IEEE Int. Conf. on Robotics and Automation*, pp. 48–55, Kobe, Japan, 2009. IEEE.

[Comaniciu 00] D. Comaniciu, V. Ramesh, P. Meer. – Real-time tracking of non-rigid objects using mean shift. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 142–149, 2000.

[Comaniciu 03] D. Comaniciu, V. Ramesh, P. Meer. – Kernel-based object tracking. *IEEE Trans. on PAMI*, 25(5):564–577, May 2003.

[Comport 03] A.I. Comport, E. Marchand, F. Chaumette. – A real-time tracker for markerless augmented reality. – *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'03*, pp. 36–45, Tokyo, Japan, octobre 2003.

[Comport 06a] A.I. Comport, E. Marchand, F. Chaumette. – Statistically robust 2d visual servoing. *IEEE Trans. on Robotics*, 22(2):415–421, apr 2006.

[Comport 06b] A.I. Comport, E. Marchand, M. Pressigout, F. Chaumette. – Real-time markerless tracking for augmented reality: the virtual visual servoing framework. *IEEE Trans. on Visualization and Computer Graphics*, 12(4):615–628, juillet 2006.

[Costa 00] M. Costa, L. Shapiro. – 3d object recognition and pose with relational indexing. *Computer Vision and Image Understanding*, 79(3):364–407, 2000.

[Coutard 11a] L. Coutard, F. Chaumette. – Visual detection and 3d model-based tracking for landing on aircraft carrier. – *IEEE Int. Conf. on Robotics and Automation, ICRA'11*, pp. 1746–1751, Shanghai, China, May 2011.

[Coutard 11b] L. Coutard, F. Chaumette, J.-M. Pflimlin. – Automatic landing on aircraft carrier by visual servoing. – *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'11*, pp. 2843–2848, San Francisco, USA, September 2011.

[Criminisi 06] A. Criminisi, G. Cross, A. Blake, V. Kolmogorov. – Bilayer segmentation of live video. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 53–60, New-York, NY, 2006.

[Cyr 01] C. Cyr, B. Kimia. – 3d object recognition using shape similiarity-based aspect graph. – *IEEE Int. Conf. on Computer Vision*, vol. 1, pp. 254–261, Vancouver, Canada, 2001. IEEE.

[Cyr 04] C.M. Cyr, B.B. Kimia. – A similarity-based aspect-graph approach to 3d object recognition. *Int. Journal of Computer Vision*, 57:5–22, 2004.

[Dalal 05] N. Dalal, B. Triggs. – Histograms of oriented gradients for human detection. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 886–893, 2005.

[Dame 10] A. Dame, E. Marchand. – Accurate real-time tracking using mutual information. – *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'10*, Seoul, Korea, October 2010.

[Dame 12] A. Dame, E. Marchand. – Second order optimization of mutual information for real-time image registration. *IEEE Trans. on Image Processing*, 21(9):4190–4203, September 2012.

[David 03] P. David, D. DeMenthon, R. Duraiswami, H. Samet. – Simultaneous pose and correspondence determination using line features. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 424–430, Madison, WI, 2003. IEEE.

[Delabarre 13] B. Delabarre, E. Marchand. – Camera localization using mutual information-based multiplane tracking. – *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'2013*, Tokyo, Japon, novembre 2013.

[deLafontaine 06] J. de Lafontaine, D. Neveu, K. Lebel. – Autonomous planetary landing using a lidar sensor: the closed-loop system. – *Guidance, Navigation and Control Systems*, vol. 606, p. 3, 2006.

[Delaune 12] J. Delaune, G. Le Esnerais, M. Sanfourche, T. Voirin, C. Bourdarias, J.L. Farges. – Optical terrain navigation for pinpoint landing: Image scale and position-guided landmark matching. *Advances in the Astronautical Sciences*, 144:627–643, 2012.

[Dementhon 95] D. Dementhon, L. Davis. – Model-based object pose in 25 lines of codes. *Int. J. of Computer Vision*, 15:123–141, 1995.

[Dhome 89] M. Dhome, M. Richetin, J.-T. Laprest, G. Rives. – Determination of the attitude of 3D objects from a single perspective view. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(12):1265–1278, décembre 1989.

[Dionnet 07] F. Dionnet, E. Marchand. – Robust stereo tracking for space robotic applications. – *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS'07*, pp. 3373–3378, San Diego, CA, October 2007.

[Drummond 02] T. Drummond, R. Cipolla. – Real-time visual tracking of complex structures. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(7):932–946, juillet 2002.

[Duda 72] R.O. Duda, P.E. Hart. – Use of the hough transformation to detect lines and curves in pictures. *Communication of the ACM*, 15:11–15, 1972.

[Dueck 07] D. Dueck, B.J. Frey. – Non-metric affinity propagation for unsupervised image categorization. – *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pp. 1–8. IEEE, 2007.

[Eggert 93] D. Eggert, K. Bowyer. – Computing the perspective projection aspect graph of solids of revolution. *IEEE Trans. on PAMI*, 15(2):109–128, février 1993.

[Elgammal 00] A. Elgammal, D. Harwood, L. Davis. – Non-parametric model for background subtraction. *European Conf. on Computer Vision*, pp. 751–767. – Dublin, Eire, Springer, 2000.

[Espiau 92] B. Espiau, F. Chaumette, P. Rives. – A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, juin 1992.

[Faugeras 93] O. Faugeras. – *Three-dimensional computer vision: a geometric viewpoint*. – MIT Press, Cambridge, Massachusetts, 1993.

[Fehse 08] W. Fehse (édité par). – *Automated rendezvous and docking of spacecraft.* – Cambridge Aerospace Series. Cambridge Univ. Press, 2008.

[Ferrari 08] V. Ferrari, L. Fevrier, F. Jurie, C. Schmid. – Groups of adjacent contour segments for object detection. – vol. 30, pp. 36–51, janvier 2008.

[Fischler 81] N. Fischler, R.C. Bolles. – Random sample consensus: A paradigm for model fitting with application to image analysis and automated cartography. *Communication of the ACM*, 24(6):381–395, juin 1981.

[Frey 07] B.J. Frey, D. Dueck. – Clustering by passing messages between data points. *Science*, 315:972–976, 2007.

[Friend 08] R. Friend. – Orbital express program summary and mission overview. – *SPIE Defense and Security Symposium*, pp. 695803–695803. Int. Society for Optics and Photonics, 2008.

[Gao 03] X.S. Gao, X.R. Hou, J. Tang, H.F. Cheng. – Complete solution classification for the perspective-three-point problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(8):930–943, 2003.

[Gavrila 99] D.M. Gavrila, V. Philomin. – Real-time object detection for "smart" vehicles. – *IEEE Int. Conf. on Computer Vision*, vol. 1, pp. 87–93, Vancouver, British Columbia, 1999. IEEE.

[Gennery 92] D.B. Gennery. – Visual tracking of known three-dimensional objects. *Int. J. of Computer Vision*, 7(3):243–270, 1992.

[Gigus 90] Z. Gigus, J. Malik. – Computing the aspect graph for line drawings of polyhedral objects. *IEEE Trans. on PAMI*, 12(2):113–122, février 1990.

[Glasner 11] D. Glasner, M. Galun, S. Alpert, R. Basri, G. Shakhnarovich. – Viewpoint-aware object detection and pose estimation. – *IEEE Int. Conf. on Computer Vision*, pp. 1275–1282, Barcelona, Spain, 2011. IEEE.

[Grelier 10] T. Grelier, P.Y. Guidotti, M. Delpech, J. Harr, J.B. Thevenet, X. Leyre. – Formation flying radio frequency instrument: first flight results from the prisma mission. – *Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing (NAVITEC), 2010 5th ESA Workshop on*, pp. 1–8. IEEE, 2010.

[Gu 10] C. Gu, X. Ren. – Discriminative mixture-of-templates for viewpoint classification. *European Conf. on Computer Vision*, pp. 408–421. – Heraklion, Crete, Springer, 2010.

[Haag 99] M. Haag, H.H. Nagel. – Combination of edge element and optical flow estimates for 3D-model-based vehicle tracking in traffic image sequences. *Int. Journal of Computer Vision*, 35(3):295–319, décembre 1999.

[Hager 98] G. Hager, K. Toyama. – The XVision system: A general-purpose substrate for portable real-time vision applications. *Computer Vision and Image Understanding*, 69(1):23–37, janvier 1998. – Also Research Report Yale University.

[Hanek 04] R. Hanek, M. Beetz. – The contracting curve density algorithm: Fitting parametric curve models to images using local self-adapting separation criteria. *Int. Journal of Computer Vision*, 59(3):233–258, 2004.

[Harris 88] C. Harris, M. Stephens. – A combined corner and edge detector. – *Alvey Conference*, pp. 147–151, Manchester, 1988.

[Harris 92] C. Harris. – Tracking with rigid objects, 1992.

[Hartley 01] R. Hartley, A. Zisserman. – *Multiple View Geometry in Computer Vision*. – Cambridge University Press, 2001.

[Hayman 03] E. Hayman, J.O. Eklundh. – Statistical background subtraction for a mobile observer. – *IEEE Int. Conf. on Computer Vision*, pp. 67–74, Vancouver, Canada, 2003. IEEE.

[Heitz 93] F. Heitz, P. Bouthemy. – Multimodal estimation of discontinuous optical flow using markov random fields. *IEEE Trans. on PAMI*, 15(12):1217–1232, décembre 1993.

[Hinterstoisser 10] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, N. Navab. – Dominant orientation templates for real-time detection of texture-less objects. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 2257–2264, San Francisco, 2010. IEEE.

[Holzer 09] S. Holzer, S. Hinterstoisser, S. Ilic, N. Navab. – Distance transform templates for object detection and pose estimation. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 1177–1184, Miami, FL, 2009. IEEE.

[Horn 87] B. Horn. – *Robot Vision*. – MIT Press, Cambridge, 1987.

[Howard 08] R. Howard, A. Heaton, R. Pinson, C. Carrington. – Orbital express advanced video guidance sensor. – *2008 IEEE Aerospace Conference*, pp. 1–10. IEEE, 2008.

[Huber 81] P.-J. Huber. – *Robust Statistics*. – Wiler, New York, 1981.

[Huttenlocher 93] D. Huttenlocher, G. Klanderman, W.J. Rucklidge. – Comparing images using the hausdorff distance. *IEEE Trans. on PAMI*, 15(9):850–863, septembre 1993.

[Inaba 00] N. Inaba, M. Oda. – Autonomous satellite capture by a space robot: world first on-orbit experiment on a japanese robot satellite ets-vii. – *IEEE Int. Conf. on Robotics and Automation*, vol. 2, pp. 1169–1174, San Francisco, CA, 2000. IEEE.

[Irani 92] M. Irani, B. Rousso, S. Peleg. – Detecting and tracking multiple moving objects using temporal integration. – *ECCV'92*, pp. 282–287, 1992.

[Isard 98] M. Isard, A. Blake. – Condensation – conditional density propagation for visual tracking. *Int. J. Computer Vision*, 29(1):5–28, janvier 1998.

[Isenberg 03] T. Isenberg, B Freudenberg, S. Schlechtweg, T. Strothotte. – A developer guide to silhouette algorithms for polygonal models. *IEEE Computer Graphics and Applications*, 23(4):28–37, 2003.

[Jain 79] R. Jain, H.-H. Nagel. – On the analysis of accumulative difference pictures from image sequences of real world scenes. *IEEE Trans. on PAMI*, 1(2):206–214, avril 1979.

[Jasiobedzki 01] P. Jasiobedzki, M. Greenspan, G. Roth. – Pose determination and tracking for autonomous satellite capture. – *Int. Symp. on Artificial Intelligence and Robotics & Automation in Space, i-SAIRAS'01,*, vol. 15, pp. 6–9, Montreal, Canada, 2001.

[Jurie 98] F. Jurie. – Tracking objects with a recognition algorithm. *Pattern Recognition Letters*, 19(3):331–340, 1998.

[Jurie 01a] F. Jurie, M. Dhome. – Real time 3D template matching. – *Int. Conf. on Computer Vision and Pattern Recognition*, vol. 1, pp. 791–796, Hawai, décembre 2001.

[Jurie 01b] F. Jurie, M. Dhome. – Real time tracking of 3d objects with occultations. – *ICIP01*, pp. I: 413–416, 2001.

[Jurie 02] F. Jurie, M. Dhome. – Hyperplane approximation for template matching. *IEEE Trans. on PAMI*, 24(7):996–1000, juillet 2002.

[Kaiser 08] C. Kaiser, F. Sjöberg, J.M. Delcura, B. Eilertsen. – SMART-OLEV—An orbital life extension vehicle for servicing commercial spacecrafts in GEO. *Acta Astronautica*, 63(1–4):400–410, 2008.

[Kanade 98] T. Kanade, R. Collins, A. Lipton, P. Burt, L. Wixson. – Advances in cooperative multi-sensor video surveillance. – *Proc. of DARPA Image Understanding Workshop*, vol. 1, p. 2, 1998.

[Karmann 90] K.-P. Karmann, A.V. Brandt, R. Gerl. – Moving object segmentation based on adaptive reference images. – *5. European Signal Processing Conference.*, vol. 2, pp. 951–954, 1990.

[Kawano 01] I. Kawano, M. Mokuno, T. Kasai, T. Suzuki. – Result of autonomous rendezvous docking experiment of engineering test satellite-vii. *Journal of Spacecraft and Rockets*, 38(1):105–111, 2001.

[Kelsey 06] J. Kelsey, J. Byrne, M. Cosgrove, S. Seereeram, R. Mehra. – Vision-based relative pose estimation for autonomous rendezvous and docking. – *2006 IEEE Aerospace Conference*, pp. 20–pp. IEEE, 2006.

[Kessler 78] D.J. Kessler, B.G. Cour-Palais. – Collision frequency of artificial satellites: The creation of a debris belt. *Journal of Geophysical Research: Space Physics (1978–2012)*, 83(A6):2637–2646, 1978.

[Kim 05] K. Kim, T. Chalidabhongse, D. Harwood, L. Davis. – Real-time foreground–background segmentation using codebook model. *Real-time imaging*, 11(3):172–185, 2005.

[Kimia 95] B. Kimia, A. Tannenbaum, S. Zucker. – Shapes, shocks, and deformations i: the components of two-dimensional shape and the reaction-diffusion space. *Int. Journal of Computer Vision*, 15(3):189–224, 1995.

[Klein 04] G. Klein, T. Drummond. – Tightly integrated sensor fusion for robust visual tracking. 22(10):769–776, 2004.

[Klein 06] G. Klein, D. Murray. – Full-3d edge tracking with a particle filter. – *British Machine Vision Conf.*, vol. 3, pp. 1119–1128, Edinburgh, septembre 2006.

[Koenderink 76] J. Koenderink, A. Van Doorn. – The singularities of the visual mapping. *Biological cybernetics*, 24(1):51–59, 1976.

[Koenderink 90] J.J. Koenderink. – Solid shape. – MIT Press, 1990.

[Koller 93] D. Koller, K. Daniilidis, H.-H. Nagel. – Model-based object tracking in monocular image sequences of road traffic scenes. *Int. Journal of Computer Vision*, 10(2):257–281, juin 1993.

[Koller 94] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, S. Russell. – Towards robust automatic traffic scene analysis in real-time. – *Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on*, vol. 1, pp. 126–131. IEEE, 1994.

[Kollnig 97] H. Kollnig, H.-H. Nagel. – 3D Pose Estimation by Directly Matching Polyhedral Models to Gray Value Gradients. *Int. Journal of Computer Vision*, 23(3):283–302, juillet 1997.

[Konrad 00] J. Konrad. – Motion detection and estimation. *Handbook of Image and Video Processing*, pp. 207–225, 2000.

[Kriegman 90] D. Kriegman, J. Ponce. – Computing exact aspect graphs of curved objects: Solids of revolution. *Int. Journal of Computer Vision*, 5(2):119–135, 1990.

[Kubota 06] T. Kubota, T. Hashimoto, J. Kawaguchi, M. Uo, K. Shirakawa. – Guidance and navigation of hayabusa spacecraft for asteroid exploration and sample return mission. – *Int. Joint Conf. SICE-ICASE*, pp. 2793–2796. IEEE, 2006.

[Kyrki 05] V. Kyrki, D. Kragic. – Integration of model-based and model-free cues for visual object tracking in 3d. – *IEEE Int. Conf. on Robotics and Automation, ICRA'05*, pp. 1566–1572, Barcelona, Spain, avril 2005.

[LaCascia 00] M. La Cascia, S. Sclaroff, V. Athitsos. – *Fast, Reliable Head Tracking under Varying Illumination: An Approach Based on Registration of Texture-Mapped 3D Models*. – Rapport de Recherche n 4, avril 2000.

[Lai 99] H. Lai, S, M. Fang. – Robust and efficient image alignment with spatially varying illumination models. – *CVPR*, pp. 2167–, 1999.

[Leibe 04] B. Leibe, A. Leonardis, B. Schiele. – Combined object categorization and segmentation with an implicit shape model. – *Workshop on Statistical Learning in Computer Vision, ECCV*, vol. 2, p. 7, 2004.

[Lepetit 06a] V. Lepetit, P. Fua. – Keypoint recognition using randomized trees. *IEEE Trans. on PAMI*, 28(9):1465–1479, septembre 2006.

[Lepetit 06b] V. Lepetit, P. Fua. – Keypoint recognition using randomized trees. *IEEE Trans. on PAMI*, 28(9):1465–1479, 2006.

[Lepetit 09] Vincent Lepetit, Francesc Moreno-Noguer, Pascal Fua. – Epnp: An accurate o (n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155–166, 2009.

[Li 93] H. Li, P. Roivainen, R. Forchheimer. – 3-d motion estimation in model-based facial image coding. *IEEE Trans. on PAMI*, 15(6):545–555, 1993.

[Lichter 04] M. Lichter, S. Dubowsky. – State, shape, and parameter estimation of space objects from range images. – *IEEE Int. Conf. on Robotics and Automation*, vol. 3, pp. 2974–2979. IEEE, 2004.

[Liebelt 08] J. Liebelt, C. Schmid, K. Schertler. – Viewpoint-independent object class detection using 3D feature maps. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, Anchorage, Alaska, juin 2008.

[Liebelt 10] J. Liebelt, C. Schmid. – Multi-view object class detection with a 3d geometric model. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 1688–1695, San Francisco, CA, June 2010.

[Lindeberg 94] T. Lindeberg. – Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of applied statistics*, 21(1-2):225–270, 1994.

[Liou 10] J.-C. Liou, Jonhson N.L., Hill N.M. – Controlling the growth of future leo debris populations with active debris removal. *Acta Astronautica*, 66(5–6):648 – 653, 2010.

[Lowe 87] D.G. Lowe. – Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–394, mars 1987.

[Lowe 91] D.G. Lowe. – Fitting parameterized three-dimensional models to images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(5):441–450, mai 1991.

[Lowe 92] D.G. Lowe. – Robust model-based motion tracking trough the integration of search and estimation. *Int. Journal of Computer Vision*, 8(2):113–122, 1992.

[Lowe 99] D.G. Lowe. – Object recognition from local scale-invariant features. – *IEEE Int. Conf. on Computer Vision*, vol. 2, pp. 1150–1157, Kerkira, Greece, 1999. Ieee.

[Lowe 01] D.G. Lowe. – Local feature view clustering for 3d object recognition. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 1, pp. I–682, Kauai, HI, 2001. IEEE.

[Lowe 04] D.G. Lowe. – Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110, 2004.

[Lu 00] C.P. Lu, G.D. Hager, E. Mjolsness. – Fast and globally convergent pose estimation from video images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(6):610–622, juin 2000.

[Lucas 81] B.D. Lucas, T. Kanade. – An iterative image registration technique with an application to stereo vision. – *Int. Joint Conf. on Artificial Intelligence, IJCAI'81*, pp. 674–679, 1981.

[Ma 04] Y. Ma, S. Soatto, J. Košecká, S. Sastry. – *An invitation to 3-D vision*. – Springer, 2004.

[Malis 98] E. Malis. – *Contributions à la modélisation et à la commande en asservissement visuel*. – PhD. Thesis, 1998.

[Marchand 99] E. Marchand, P. Bouthemy, F. Chaumette, V. Moreau. – Robust real-time visual tracking using a 2D-3D model-based approach. – *IEEE Int. Conf. on Computer Vision, ICCV'99*, vol. 1, pp. 262–268, Kerkira, Greece, septembre 1999.

[Marchand 02] E. Marchand, F. Chaumette. – Virtual visual servoing: a framework for real-time augmented reality. – G. Drettakis, H.-P. Seidel (édité par), *EUROGRAPHICS'02 Conf. Proceeding*, vol. 21(3) of *Computer Graphics Forum*, pp. 289–298, Saarebrücken, Germany, septembre 2002.

[Marchand 05] E. Marchand, F. Spindler, F. Chaumette. – ViSP for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine*, 12(4):40–52, décembre 2005.

[Martin 13] Th Martin, E Pérot, M-Ch Desjean, L Bitetti. – Active debris removal mission design in low earth orbit. – *Progress in Propulsion Physics*, vol. 4, pp. 763–788. EDP Sciences, 2013.

[Masson 03] L. Masson, F. Jurie, M. Dhome. – Contour/texture approach for visual tracking. – *13th Scandinavian Conf. on Image Analysis, SCIA 2003*, vol. 2749 of *Lecture Notes in Computer Science*, pp. 661–668. Springer, 2003.

[Matas 04] J. Matas, O. Chum, M. Urban, T. Pajdla. – Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22(10):761–767, octobre 2004.

[Mémin 02] E. Mémin, P. Pérez. – Hierarchical estimation and segmentation of dense motion fields. *Int. Journal of Computer Vision*, 46(2):129–155, 2002.

[Mikolajczyk 02] K. Mikolajczyk, C. Schmid. – An affine invariant interest point detector. *European Conf. on Computer Vision*, pp. 128–142. – Copenhagen, Denmark, Springer, 2002.

[Miravet 08] C. Miravet, L. Pascual, E. Krouch, J.M. del Cura. – An image-based sensor system for autonomous rendez-vous with uncooperative satellites. 2008.

[Mittal 00] A. Mittal, D. Huttenlocher. – Scene modeling for wide area surveillance and image synthesis. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 160–167, Hilton Head, 2000. IEEE.

[Mühlbauer 12] Q. Mühlbauer, L. Richter, C. Kaiser, P. Hofmann. – Robotics space systems and subsystems for advanced future programmes. – *Int. Symp. on Artificial Intelligence, Robotics and Automation in Space, i-SAIRAS 2012*, pp. 4–6, Torino, Italy, Sept, 2012.

[NASA 10] NASA. – On-orbit satellite servicing study. *http://ssco.gsfc.nasa.gov/*, NASA Space Servicing Capabilities Project, Goddard Space Flight Center, 2010.

[NASA 12] NASA. – Orbital debris management and risk mitigation. *http://www.nasa.gov/offices/oce/appel/knowledge/publications/appel-releases-ibook.html*, NASA Academy of Program Project and Engineering Leadership (APPEL), 2012.

[Nelson 91] R.C. Nelson. – Qualitative detection of motion by a moving observer. *International journal of computer vision*, 7(1):33–46, 1991.

[Odobez 97] J.-M. Odobez, P. Bouthemy. – Separation of moving regions from background in an image sequence acquired with a mobil camera. *Video Data Compression for Multimedia Computing*, pp. 283–311. – Springer, 1997.

[Olson 97] C.F. Olson, D.P. Huttenlocher. – Automatic target recognition by matching oriented edge pixels. *IEEE Trans. on Image Processing*, 6(1):103–113, 1997.

[Oumer 12a] N. Oumer, G. Panin. – 3d point tracking and pose estimation of a space object using stereo images. – *IAPR Int. Conf. on Pattern Recognition*, pp. 796–800. IEEE, 2012.

[Oumer 12b] N. Oumer, G. Panin. – Tracking and pose estimation of non-ccoperative satellite for on-orbit servicing. 2012.

[Ozuysal 07] M. Ozuysal, P. Fua, V. Lepetit. – Fast keypoint recognition in ten lines of code. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, 2007.

[Ozuysal 09] M. Ozuysal, V. Lepetit, P. Fua. – Pose Estimation for Category Specific Multiview Object Localization. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, Miami, Fl, 2009.

[Ozuysal 10] M. Ozuysal, M. Calonder, V. Lepetit, P. Fua. – Fast keypoint recognition using random ferns. *IEEE Trans. on PAMI*, 32(3):448–461, mars 2010.

[Panin 06] G. Panin, A. Ladikos, A. Knoll. – An efficient and robust real-time contour tracking system. – *IEEE Int. Conf. on Computer Vision Systems, ICVS'06*, pp. 44–44. IEEE, 2006.

[Panin 08a] G. Panin, A. Knoll. – Mutual information-based 3d object tracking. *Int. Journal of Computer Vision*, 78(1):107–118, 2008.

[Panin 08b] G. Panin, E. Roth, A. Knoll. – Robust contour-based object tracking integrating color and edge likelihoods. – *Proc. of the Vision, Modeling, and Visualization Conference*, pp. 227–234, Konstanz, Germany, octobre 2008.

[Payet 11] N. Payet, S. Todorovic. – From contours to 3d object detection and pose estimation. – *IEEE Int. Conf. on Computer Vision*, pp. 983–990, Barcelona, Spain, 2011.

[Persoon 77] E. Persoon, K. Fu. – Shape discrimination using fourier descriptors. *IEEE Trans. on Systems, Man, and Cybernetics*, 7(3):170–179, 1977.

[Persson 06] S. Persson, P. Bodin, E. Gill, J. Harr, J. Jörgensen. – Prisma–an autonomous formation flying mission. – *ESA Small Satellite Systems and Services Symposium (4S), Sardinia, Italy*, pp. 25–29, 2006.

[Petit 11] A. Petit, E. Marchand, K. Kanani. – Vision-based space autonomous rendezvous : A case study. – *IEEE/RSJ Int. Conf on Intelligent Robots and Systems, IROS'2011*, pp. 619–624, San Francisco, USA, September 2011.

[Pinard 07] D. Pinard, S. Reynaud, P. Delpy, S.E. Strandmoe. – Accurate and autonomous navigation for the ATV. *Aerospace Science and Technology*, 11(6):490 – 498, 2007.

[Poberezhskiy 12] I. Poberezhskiy, A. Johnson, D. Chang, E. Ek, D. Natzic, G. Spiers, S. Penniman, B. Short. – Flash lidar performance testing: configuration and results. – *SPIE Defense, Security, and Sensing*, pp. 837905–837905. International Society for Optics and Photonics, 2012.

[Pressigout 04] M. Pressigout, E. Marchand. – Model-free augmented reality by virtual visual servoing. – *IAPR Int. Conf. on Pattern Recognition, ICPR'04*, vol. 2, pp. 887–891, Cambridge, UK, août 2004.

[Pressigout 06a] M. Pressigout, E. Marchand. – Fusion de primitives visuelles pour le suivi 3d temps-rel. – *15e congrs francophone AFRIF-AFIA Reconnaissance des Formes et Intelligence Artificielle, RFIA'06*, Tours, France, jan 2006.

[Pressigout 06b] M. Pressigout, E. Marchand. – Real-time 3d model-based tracking: Combining edge and texture information. – *IEEE Int. Conf. on Robotics and Automation, ICRA'06*, pp. 2726–2731, Orlando, Florida, mai 2006.

[Pressigout 07] M. Pressigout, E. Marchand. – Real-time hybrid tracking using edge and texture information. *Int. Journal of Robotics Research, IJRR*, 26(7):689–713, juillet 2007.

[Pressigout 08] M. Pressigout, E. Marchand, E. Mémin. – Hybrid tracking approach using optical flow and pose estimation. – *IEEE Int. Conf. on Image Processing, ICIP'08*, pp. 2720–2723, San Diego, California, October 2008.

[Prisacariu 12] V. Prisacariu, I. Reid. – Pwp3d: Real-time segmentation and tracking of 3d objects. *Int. Journal of Computer Vision*, 98(3):335–354, 2012.

[Prisacariu 13a] V.A. Prisacariu, O. Kähler, D.W. Murray, I. Reid. – Simultaneous 3d tracking and reconstruction on a mobile phone. *IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'13*. – Springer, 2013.

[Prisacariu 13b] V.A. Prisacariu, A.V. Segal, I. Reid. – Simultaneous monocular 2d segmentation, 3d pose recovery and 3d reconstruction. *Computer Vision–ACCV 2012*, pp. 593–606. – Springer, 2013.

[Pundlik 06] S. Pundlik, S. Birchfield. – Motion segmentation at any speed. – *British Machine Vision Conf.*, pp. 427–436, 2006.

[Rabiner 89] L.R. Rabiner. – A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[Rank 11] P. Rank, Q. Mühlbauer, W. Naumann, K. Landzettel. – The deos automation and robotics payload. – *Symp. on Advanced Space Technologies in Robotics and Automation, ASTRA*, the Netherlands, 2011.

[Reinbacher 10] C. Reinbacher, M. Ruether, H. Bischof. – Pose estimation of known objects by efficient silhouette matching. – *IAPR Int. Conf. on Pattern Recognition*, 2010.

[Reitmayr 06] G. Reitmayr, T. Drummond. – Going out: robust model-based tracking for outdoor augmented reality. – *IEEE/ACM Int. Symp. on Mixed and Augmented Reality, ISMAR'2006*, pp. 109–118, Santa Barbara, CA, octobre 2006.

[Ren 03] Y. Ren, C.-S. Chua, Y.-K. Ho. – Statistical background modeling for non-stationary camera. *Pattern Recognition Letters*, 24(1):183–196, 2003.

[Richa 11] R. Richa, R. Sznitman, R. Taylor, G. Hager. – Visual tracking using the sum of conditional variance. – pp. 2953–2958. IEEE, 2011.

[Rodrigues 12] J. Rodrigues, J.-S. Kim, M. Furukawa, J. Xavier, P. Aguiar, T. Kanade. – 6d pose estimation of textureless shiny objects using random ferns for bin-picking. – pp. 3334–3341, Algarve, Portugal, 2012. IEEE.

[Rosten 05] E. Rosten, T. Drummond. – Fusing points and lines for high performance tracking. – *IEEE Int. Conf. on Computer Vision*, vol. 2, pp. 1508–1515, Beijing, China, 2005.

[Rother 04] C. Rother, V. Kolmogorov, A. Blake. – Grabcut: Interactive foreground extraction using iterated graph cuts. – *ACM Transactions on Graphics (TOG)*, vol. 23, pp. 309–314. ACM, 2004.

[Rowe 96] S. Rowe, A. Blake. – Statistical mosaics for tracking. *Image and Vision Computing*, 14(8):549–564, 1996.

[Rucklidge 95] W.J. Rucklidge. – Locating objects using the hausdorff distance. – *IEEE Int. Conf. on Computer Vision*, pp. 457–464, Boston, Massachusetts, 1995. IEEE.

[Ruel 05] S. Ruel, C. English, M. Anctil, P. Church. – 3dlasso: Real-time pose estimation from 3d data for autonomous satellite servicing. – *Int. Symp. on Artificial Intelligence for Robotics and Automation in Space, i-SAIRAS*, Munich, Germany, 2005.

[Ruel 08] S. Ruel, D. Ouellet, T. Luu, D. Laurendeau. – Automatic tracking initialization from tridar data for autonomous rendezvous & docking. – *Proc. of the iSAIRAS 2008 conference*, Hollywood, CA, 2008.

[Ruel 10] S. Ruel, T. Luu, A. Berube. – On-orbit testing of target-less tridar 3d rendezvous and docking sensor. – *Proc. of 2010 iSAIRAS conference*, Sapporo, Japan, 2010.

[Samson 04] C. Samson, C. English, A. Deslauriers, I. Christie, F. Blais, F. Ferrie. – Neptec 3d laser camera system: From space mission sts-105 to terrestrial applications. – *2002 ASTRO Conference*, 2004.

[Savarese 07] S. Savarese, L. Fei-Fei. – 3d generic object categorization, localization and pose estimation. – *IEEE Int. Conf. on Computer Vision*, Rio de Janeiro, Brazil, October 2007.

[Schlkopf 02] B. Schlkopf, A. Smola. – Learning with kernels: support vector machines, regularization, optimization, and beyond. *The MIT Press*, 2002.

[Sebastian 01a] T. Sebastian, P. Klein, B. Kimia. – Alignment-based recognition of shape outlines. *Visual Form 2001*, pp. 606–618. – Springer, 2001.

[Sebastian 01b] T. Sebastian, P. Klein, B. Kimia. – Recognition of shapes by editing shock graphs. – *IEEE Int. Conf. on Computer Vision*, vol. 1, pp. 755–762, Vancouver, Canada, 2001.

[Sellmaier 10] F. Sellmaier, T. Boge, J. Spurmann, S. Gully, T. Rupp, F. Huber. – On-orbit servicing missions: Challenges and solutions for spacecraft operations. 2010.

[Sheikh 05] Y. Sheikh, M. Shah. – Bayesian modeling of dynamic scenes for object detection. *IEEE Trans. on PAMI*, 27(11):1778–1792, novembre 2005.

[Sheikh 09] Y. Sheikh, O. Javed, T. Kanade. – Background subtraction for freely moving cameras. – *IEEE Int. Conf. on Computer Vision*, pp. 1219–1225, Kyoto, Japan, 2009.

[Shi 94] J. Shi, C. Tomasi. – Good features to track. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'94*, pp. 593–600, Seattle, Washington, juin 1994.

[Shimshoni 97] I. Shimshoni, J. Ponce. – Finite-resolution aspect graphs of polyhedral objects. *IEEE Trans. on PAMI*, 19(4):315–327, avril 1997.

[Shotton 05] J. Shotton, A. Blake, R. Cipolla. – Contour-based learning for object detection. – *IEEE Int. Conf. on Computer Vision*, pp. 503–510, 2005.

[Srivastava 05] A. Srivastava, S. H Joshi, W. Mio, X. Liu. – Statistical shape analysis: Clustering, learning, and testing. *IEEE Trans. on PAMI*, 27(4):590–602, avril 2005.

[Stark 10] M. Stark, M. Goesele, B. Schiele. – Back to the future: Learning shape models from 3d cad data. – *British Machine Vision Conf.*, Aberystwyth, Wales, août 2010.

[Stauffer 99] C. Stauffer, E. Grimson. – Adaptive background mixture models for real-time tracking. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 2246–2252, Miami, Fl, 1999.

[Steger 02] C. Steger. – Occlusion, clutter, and illumination invariant object recognition. *Int. Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(3/A):345–350, 2002.

[Stein 97] G.P. Stein. – Lens distortion calibration using point correspondences. – *IEEE Int. Conf on computer Vision and Pattern Recognition, CVPR97*, pp. 602–608, 1997.

[Stewman 88] J. Stewman, K. Bowyer. – Creating the perspective projection aspect graph of polyhedral objects. – *IEEE Int. Conf. on Computer Vision*, pp. 494–500. IEEE, 1988.

[Strandmoe 08] S. Strandmoe, E. DePasquale, I. Escane, M. Augelli, G. Personne, B. Cavrois, N. Fau, M. Yu, M. Zink, X. Clerc et al. – Automated transfer vehicle (atv) flight control achievements. – *7th Int. ESA Conference on Guidance, Navigation & Control Systems*, 2008.

[Strzodka 03] R. Strzodka, I. Ihrke, M. Magnor. – A graphics hardware implementation of the generalized hough transform for fast object recognition, scale, and 3d pose detection. – *Int. Conf. on Image Analysis and Processing*, pp. 188–193. IEEE, 2003.

[Su 09] H. Su, M. Sun, L. Fei-Fei, S. Savarese. – Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. – *IEEE Int. Conf. on Computer Vision*, Kyoto, Japan, October 2009.

[Sun 09] M. Sun, H. Su, S. Savarese, L. Fei-Fei. – A multi-view probabilistic model for 3d object classes. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 1247–1254, Miami, FL, 2009. IEEE.

[Tamaazousti 11] M. Tamaazousti, V. Gay-Bellile, S.N. Collette, S. Bourgeois, M. Dhome. – Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. – *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 3073–3080. IEEE, 2011.

[Tamadazte 10] Brahim Tamadazte, Eric Marchand, Sounkalo Dembélé, Nadine Le Fort-Piat. – Cad model-based tracking and 3d visual-based control for mems microassembly. *The International Journal of Robotics Research*, 29(11):1416–1434, 2010.

[Teulière 09] C. Teulière, E. Marchand, L. Eck. – A combination of particle filtering and deterministic approaches for multiple kernel tracking. – *IEEE Int. Conf. on Robotics and Automation*, pp. 3948–3954. IEEE, 2009.

[Teulière 10] C. Teulière, E. Marchand, L. Eck. – Using multiple hypothesis in model-based tracking. – *IEEE Int. Conf. on Robotics and Automation, ICRA'10*, pp. 4559–4565, Anchorage, Alaska, mai 2010.

[Thomas 06] A. Thomas, V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, L. Van Gool. – Towards multi-view object class detection. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 1589–1596, New York, NY, 2006. IEEE.

[Tingdahl 10] D. Tingdahl, M. Vergauwen, L. Van Gool. – Harvd image processing algorithms detailed design. – 2010. EADS Astrium.

[Tola 08] E. Tola, V. Lepetit, P. Fua. – A fast local descriptor for dense matching. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 1–8, Anchorage, Alaska, 2008. IEEE.

[Tomasi 91] C. Tomasi, T. Kanade. – *Detection and Tracking of Point Features.* – Rapport de Recherche nCMU-CS-91-132, Carnegie Mellon University Technical Report, avril 1991.

[Toshev 09] A. Toshev, A. Makadia, K. Daniilidis. – Shape-based object recognition in videos using 3d synthetic object models. 0:288–295, 2009.

[Tsai 86] R.Y. Tsai. – An efficient and accurate camera calibration technique for 3D machine vision. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'86*, pp. 364–374, Miami, Floride, juin 1986.

[Tsai 87] R.Y. Tsai, R. Lenz. – *A New Technique for Autonomous and Efficient 3D Robotics Hand-Eye Calibration*. – Rapport de Recherche n RC12212, Yorktown Heights, New York, IBM T.J. Watson Research Center, octobre 1987.

[Turk 91] M. Turk, A. Pentland. – Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991.

[Tweddle 10] B. Tweddle. – *Computer vision based navigation for spacecraft proximity operations*. – PhD. Thesis, Massachusetts Institute of Technology, 2010.

[Ulrich 09] M. Ulrich, C. Wiedemann, C. Steger. – Cad-based recognition of 3d objects in monocular images. – *IEEE Int. Conf. on Robotics and Automation*, pp. 2090–2097, 2009.

[Vacchetti 03] L. Vacchetti, V. Lepetit, P. Fua. – Stable 3–d tracking in real-time using integrated context information. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition, CVPR'03*, vol. 2, pp. 241–248, Madison, WI, juin 2003.

[Vacchetti 04a] L. Vacchetti, V. Lepetit, P. Fua. – Combining edge and texture information for real-time accurate 3d camera tracking. – *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'04*, pp. 48–57, Arlington, VA, novembre 2004.

[Vacchetti 04b] L. Vacchetti, V. Lepetit, P. Fua. – Combining edge and texture information for real-time accurate 3d camera tracking. – *ACM/IEEE Int. Symp. on Mixed and Augmented Reality, ISMAR'2004*, vol. 2, pp. 48–57, Arlington, Va, novembre 2004.

[Vacchetti 04c] L. Vacchetti, V. Lepetit, P. Fua. – Stable real-time 3d tracking using online and offline information. *IEEE Trans. on PAMI*, 26(10):1385–1391, October 2004.

[VanPham 12] B. Van Pham, S. Lacroix, M. Devy. – Vision-based absolute navigation for descent and landing. *Journal of Field Robotics*, 29(4):627–647, 2012.

[Viola 97] P. Viola, W. Wells. – Alignment by maximization of mutual information. *Int. Journal of Computer Vision*, 24(2):137–154, 1997.

[W. 05] Quen-Zong W., H.-Y. Cheng, B.-S. Jeng. – Motion detection via change-point detection for cumulative histograms of ratio images. *Pattern Recognition Letters*, 26(5):555–563, 2005.

[Woffinden 07] D.C. Woffinden, D.K. Geller. – Navigating the road to autonomous orbital rendezvous. *Journal of Spacecraft and Rockets*, 44(4):898–909, 2007.

[Woffinden 08] D.C. Woffinden. – *Angles-only navigation for autonomous orbital rendezvous*. – ProQuest, 2008.

[Wuest 07] H. Wuest, D. Stricker. – Tracking of industrial objects by using cad models. *Journal of Virtual Reality and Broadcasting*, 4(1), April 2007.

[Xiao 03] J. Xiao, M. Shah. – Two-frame wide baseline matching. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 603–609, Madison, WI, 2003. IEEE.

[Xiao 05a] J. Xiao, M. Shah. – Accurate motion layer segmentation and matting. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, vol. 2, pp. 698–703, San Diego, 2005.

[Xiao 05b] J. Xiao, M. Shah. – Motion layer extraction in the presence of occlusion using graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1644–1659, 2005.

[Yan 07] P. Yan, S. Khan, M. Shah. – 3d model based object class detection in an arbitrary view. – *IEEE Int. Conf. on Computer Vision*, pp. 1–6, Rio de Janeiro, Brazil, 2007.

[Yano 06] H. Yano, T. Kubota, H. Miyamoto, T. Okada, D. Scheeres, Y. Takagi, K. Yoshida, M. Abe, S. Abe, O. Barnouin-Jha et al. – Touchdown of the hayabusa spacecraft at the muses sea on itokawa. *Science*, 312(5778):1350–1353, 2006.

[Yin 07] P. Yin, A. Criminisi, J. Winn, M. Essa. – Tree-based classifiers for bilayer video segmentation. – *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pp. 1–8, Minneapolis, Minnesota, 2007. IEEE.

[Yoon 08] Y. Yoon, A. Kosaka, A.C. Kak. – A new kalman-filter-based framework for fast and accurate visual tracking of rigid objects. *IEEE Trans. on Robotics*, 24(5):1238–1251, octobre 2008.

[Zhang 95] Z. Zhang, R. Deriche, O. Faugeras, Q.-T. Luong. – A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78:87–119, octobre 1995.

[Zhu 96] S. Zhu, A. Yuille. – Forms: a flexible object recognition and modelling system. *Int. Journal of Computer Vision*, 20(3):187–212, 1996.

[Zia 11] Z. Zia, M. Stark, K. Schindler, S. Bernt. – Revisiting 3d geometric models for accurate object shape and pose. – *IEEE Workshop on 3D Representation and Recognition*, Barcelona, Spain, novembre 2011.

[Zivkovic 04] Z. Zivkovic. – Improved adaptive gaussian mixture model for background subtraction. – *IAPR Int. Conf. on Pattern Recognition*, pp. 28–31, 2004.

Abstract

In this thesis, we address the issue of fully localizing a known object through computer vision, using a monocular camera, what is a central problem in robotics.A particular attention is here paid on space robotics applications, with the aims of providing a unified visual localization system for autonomous navigation purposes for space rendezvous and proximity operations. Two main challenges of the problem are tackled: initially detecting the targeted object and then tracking it frame-by-frame, providing the complete pose between the camera and the object, knowing its 3D CAD model. For detection, the pose estimation process is based on the segmentation of the moving object and on an efficient probabilistic edge-based matching and alignment procedure of a set of synthetic views of the object with a sequence of initial images. For the tracking phase, pose estimation is handled through a 3D model-based tracking algorithm, for which we propose three types of visual features, pertinently representing the object with its edges, its silhouette and with a set of interest points. The reliability of the localization process is evaluated by propagating the uncertainty from the errors of the visual features. This uncertainty besides feeds a linear Kalman filter on the camera velocity parameters. Qualitative and quantitative experiments have been performed on various synthetic and real data, with challenging imaging conditions, showing the efficiency and the benefits of the different contributions, and their compliance with space rendezvous applications.

Keywords : Visual tracking, object detection, moving object segmentation, space robotics

Résumé

Dans cette thèse nous traitons le problème de localiser un object connu par vision artificielle, de manière complète, précise et intègre, en utilisant une caméra monoculaire, ce qui constitue un problème majeur dans des domaines comme la robotique. L'attention est ici portée sur des applications de robotique spatiale, dans le but de concevoir un système de localisation visuelle pour des operations de rendezvous spatial autonome. Deux aspects principaux du problème sont abordés: celui de la localisation initiale de l'objet ciblé, puis celui du suivi de cet objet image par image, donnant la pose complète entre la caméra et l'objet, connaissant son modèle 3D. Pour la détection, l'estimation de pose est basée sur une segmentation de l'objet en mouvement et sur une procédure probabiliste d'appariement et d'alignement basée contours de vues synthétiques de l'objet avec une séquence d'images initiales. Pour la phase de suivi, l'estimation de pose repose sur un algorithme de suivi basé modèle 3D, pour lequel nous proposons trois types de primitives visuelles, dans l'idée de décrire l'objet considéré par ses contours, sa silhouette et par un ensemble de points d'intérêts. L'intégrité du système de localisation est evaluée en propageant l'incertitude sur les primitives visuelles. Cette incertitude est par ailleurs utilisée au sein d'un filtre de Kalman sur les paramètres de vitesse. Des tests qualitatifs et quantitatifs ont été réalisés, sur des données synthétiques et réelles, avec notamment des conditions d'image difficiles, montrant ainsi l'efficacité et les avantages des différentes contributions proposées, et leur validité dans un contexte de rendezvous spatial.

Mots clefs : Suivi visuel, detection d'objets, segmentation, robotique spatiale