SAPIENZA
UNIVERSITÀ DI ROMA

Vision-based methods for state estimation and control of robotic systems with application to mobile and surgical robots

Dottorato di Ricerca in Automatica, Bioingegneria e Ricerca Operativa (ABRO) – XXXI Ciclo

Candidate

Marco Ferro

ID number 1575233

Thesis Advisor

Prof. Marilena Vendittelli

A thesis submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Automatica, Bioingegneria e Ricerca Operativa (ABRO)

February 2019

Thesis defended on 26th February 2019
in front of a Board of Examiners composed by:

Prof. Giovanni Ulivi (chairman)

Prof. Giancarlo Bigi

Prof. Giovanni Sparacino

Vision-based methods for state estimation and control of robotic systems with application to mobile and surgical robots

Ph.D. thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: ferro@diag.uniroma1.it

Abstract

For autonomous systems that need to perceive the surrounding environment for the accomplishment of a given task, vision is a highly informative exteroceptive sensory source. When gathering information from the available sensors, in fact, the richness of visual data allows to provide a complete description of the environment, collecting geometrical and semantic information (e.g., object pose, distances, shapes, colors, lights). The huge amount of collected data allows to consider both methods exploiting the totality of the data (*dense* approaches), or a reduced set obtained from feature extraction procedures (*sparse* approaches). This manuscript presents dense and sparse vision-based methods for control and sensing of robotic systems. First, a safe navigation scheme for mobile robots, moving in unknown environments populated by obstacles, is presented. For this task, dense visual information is used to perceive the environment (i.e., detect ground plane and obstacles) and, in combination with other sensory sources, provide an estimation of the robot motion with a linear observer. On the other hand, sparse visual data are extrapolated in terms of geometric primitives, in order to implement a visual servoing control scheme satisfying proper navigation behaviours. This controller relies on visual estimated information and is designed in order to guarantee safety during navigation. In addition, redundant structures are taken into account to re-arrange the internal configuration of the robot and reduce its encumbrance when the workspace is highly cluttered.

Vision-based estimation methods are relevant also in other contexts. In the field of surgical robotics, having reliable data about unmeasurable quantities is of great importance and critical at the same time. In this manuscript, we present a Kalman-based observer to estimate the 3D pose of a suturing needle held by a surgical manipulator for robot-assisted suturing. The method exploits images acquired by the endoscope of the robot platform to extrapolate relevant geometrical information and get projected measurements of the tool pose. This method has also been validated with a novel simulator designed for the da Vinci robotic platform, with the purpose to ease interfacing and employment in ideal conditions for testing and validation.

The Kalman-based observers mentioned above are classical *passive* estimators, whose system inputs used to produce the proper estimation are theoretically arbitrary. This does not provide any possibility to *actively* adapt input trajectories in order to *optimize* specific requirements on the performance of the estimation. For this purpose, *active estimation* paradigm is introduced and some related strategies are presented. More specifically, a novel active sensing algorithm employing visual dense information is described for a typical Structure-from-Motion (SfM) problem. The algorithm generates an optimal estimation of a scene observed by a moving camera, while minimizing the maximum uncertainty of the estimation. This approach can be applied to any robotic platforms and has been validated with a manipulator arm equipped with a monocular camera.

A Fede.

Ringraziamenti

Sono stati tre anni intensi. Se è vero che il Dottorato è stato finora il periodo più impegnativo della mia vita, qualche saggio ma pessimista interlocutore potrebbe ribattere che il periodo più difficile è sempre il prossimo.
Oggi però non c'è spazio per il pessimismo, perché il valore simbolico di questa giornata ruba un po' la scena al resto. Per quanto possa suonare retorico, quello di oggi, infatti, è l'atto finale di un lungo capitolo che ha segnato profondamente la mia crescita formativa ed umana, rendendomi una persona molto diversa, più forte, di quella che tre anni fa si affacciava impaurita alla finestra di questo percorso. Ecco, non potevo farlo suonare più retorico, eppure vi giuro che c'è un fondo di verità. Sento, per questo, la necessità di portare un pensiero o un ringraziamento alle persone che sono state protagoniste di questa tappa.
Il primo ringraziamento va alla Prof. Marilena Vendittelli, per la sua guida costante, per i suoi consigli, per la sua totale disponibilità e per la sua pazienza. Oltre che la supervisor del mio percorso di Dottorato, è stata una figura di riferimento fondamentale per la forza, la tenacia, la passione e la dedizione al lavoro, unite ad una gentilezza e ad un'umanità non scontate.
Grazie ad Antonio e ad Andrea per il loro supporto remoto anche nelle ore più scomode della giornata, per i preziosi consigli e gli stimolanti confronti avuti nello sviluppo di un lavoro che può quasi avere fine (cross fingers!).
Grazie al gruppo di ricerca del Prisma Lab e del Centro Icaros: grazie ad Andrea, Mario, Fanny e al Prof. Bruno Siciliano. Poter collaborare con dei ricercatori di altissimo profilo, per di più riavvicinandomi a casa e alla mia Napoli, è stato un onore e una soddisfazione doppia. Ho imparato tanto e spero che possano esserci altre collaborazioni in futuro.
Grazie al gruppo di ricerca di Rennes e a tutto il team Rainbow: grazie a Paolo Robuffo Giordano per avermi ospitato all'IRISA, per avermi offerto la possibilità di lavorare su argomenti estremamente stimolanti e di grande interesse, e soprattutto per l'enorme pazienza dimostrata. Grazie anche a Paolo Salaris per il supporto a distanza fornito mentre ero lì.
Grazie a Maren Bennewitz e François Chaumette, per aver accettato il ruolo di revisori della mia tesi di Dottorato. A François, in particolare, porto un ringraziamento affettuoso per i bei ricordi e le stimolanti conversazioni avute durante il mio soggiorno a Rennes.

Ringrazio mia madre e mio padre, perché il supporto, l'incoraggiamento e l'amore dimostrati ogni singolo giorno sono stati decisivi perché potessi arrivare fino in fondo. Sono stati i motivatori costanti del mio percorso di crescita, i punti di riferimento dei

valori con cui ho vissuto quest'esperienza e vivrò le successive. Grazie dell'appoggio totale ed incondizionato ad un figlio che potrebbe farsi sentire più spesso su Skype. Grazie a Federica. Potremmo chiuderla così, consapevoli del fatto che queste righe, questa tesi, questa giornata speciale, tutto quanto, adesso non ci sarebbero stati senza di lei, e il mio Dottorato si sarebbe probabilmente fermato al terzo giorno. È il motore che ha mosso tutto, che mi ha incoraggiato, spronato, cazziato, sollevato, amato. È la mia compagna, la persona che con me è cresciuta, mi ha fatto crescere e continua a farlo tutt'ora. È la luce che mi ha illuminato la strada nei momenti più bui e ha contribuito a rendermi l'uomo che sono ora. Grazie.
Grazie alla mia famiglia, che anche da lontano fa arrivare il suo supporto e il suo affetto. Grazie a zio Ercolino e zia Maria, a Carmen, Tina e Giusy, perché mi fanno sentire il loro abbraccio caloroso anche se sono diventato un desaparecido.
Grazie a Giovanna, perchè da brava sorella è stata un punto di riferimento per la forza e la caparbietà proprie di chi getta il cuore oltre l'ostacolo, per il coraggio e la voglia di non arrendersi. Sarà un onore essere lì a *testimoniare* e dire "Sì allora io c'ero, mo ti spiego ... ".
Grazie a Bart. Negli ultimi ringraziamenti l'avevo chiamato "commilitone sul campo di battaglia". Adesso abbiamo combattuto in due raggruppamenti militari diversi, ma la nostra originale intimità da caserma non è mai andata perduta.
Grazie a Fabrizio, Ettore, Valeria, Francesco, per aver alleggerito le mie giornate romane con serene e distensive partite a Risiko.
Grazie ai "ragazzi del DIAG": a Daniele, Nicola, Paolo, Valerio, Claudio, Emanuele, Massimo, Barbara, Giulio, Spyros. Per aver condiviso stress, ansie e scleri annessi.
Grazie agli "amici di MARR" sparsi un po' in giro per l'Italia e per il mondo, ma sempre presenti: grazie a Cifo, Simone, Alessandro, Valerio, Antonio, Andrea, Roberto, Tommaso, Maria Rita.
Grazie a Fabrizio, Claudio, Marco Aggravi, Giuseppe per aver reso solida la mia esperienza formativa a Rennes e per avermi aiutato ad ambientarmi. Ho lasciato a Rennes un pezzo di cuore e un po' di casa.
A proposito di pezzi e di casa: grazie a Marco Cognetti, per tante, tante cose. Potrei dire perché attualmente custodisce - ancora - un mio pezzo di casa identificato in un piumone, qualche gruccia e sicuramente qualcos'altro che non ricordo, in uno scatolone che prima o poi - giuro - mi farò rispedire indietro, sempre che nel frattempo non abbia venduto tutto al mercato di Rennes. In realtà è un grazie per essere l'amico che è, per essere stato un fratello maggiore, pronto ad aiutarmi in qualunque momento, ovunque fossi, per ogni necessità. Per avermi ospitato ad oltranza a casa condividendo con me le sue lenzuola, per avermi incoraggiato, per avermi supportato, perché c'è stato in tutte le mie difficoltà.
Grazie anche a Flavia, Riccardo, Khaled, Ida, Fausta, Rosa, Chiara e a tutti gli altri ragazzi di Rennes con cui ho condiviso le giornate ed un'esperienza davvero unica.
E poi per finire, last but not least, menzione speciale finale per loro, che ci sono sempre. Possono cambiare le stagioni, invertirsi i poli, e anche se si sono trasferiti in giro per il mondo rispetto a tre anni fa, loro ci sono. Se penso ad un esempio concreto dell'idea di Amicizia, non posso che pensare a loro. Mi auguro di averli sempre accanto, per continuare a condividere gli anni, le esperienze e gli eventi che verranno, da qui a quando faremo a gara a chi ha ancora più denti. Grazie Arcangelo, Francesco, Mario, Pina, Sara.

Contents

Chapter 1

Introduction

In the sensory nervous system of animals, vision typically represents the richest source of information about the description and the understanding of the surrounding environment. The huge amount of data processed by the visual system, in fact, enables to accomplish a variety of complex tasks to provide a comprehensive depiction of the phenomena experienced around. In particular, the interconnection between the visual system and the brain, through the visual cortex, additionally allows to generate a semantic knowledge from the processed data, as well as infer further information, e.g., about classification of objects, estimation of distances and movements (see Figure 1.1). As a result, these animals develop skills such as sense of orientation, self-localization and capability to plan actions and infer consequences while moving, based on what they currently observe through their eyes.

Nature offers several examples of such vision-based behaviors: ants combine head and body motion to scan briefly the environment, in order to localize themselves with respect to the nest, and move on visually guided paths to search food [1][2]. Honeybees exploit the apparent motion of the optical flow perceived by left and right images to estimate distances, avoid collisions and assist landing on edges [3]. Shrimps use photoreceptors ultraviolet, spatial and color vision and can adapt their photoreception capabilities based on the habitat and the behavior [4]. Human beings are able to accomplish very complex visually guided behaviors through the stereoscopic visual system, that is decisive for semantic reasoning, action planning and inference of effects and consequences on the environment. All these examples show that vision actually represents a fundamental component to assist and guarantee autonomy and survival in non-deterministic, highly complex and dynamic environments.

Figure 1.1. The human beings process visual information through the visual cortex[1].

As trusty *replica* of animals and human beings, modern robotic platforms are programmed deliberately to behave analogously and with similar performances. The complete sensory equipment, mounted on the robot platform, is an artificial representation

[1]Retrieved from `https://discoveryeye.org/optic-nerve-visual-link-brain/`

of the sensory nervous system of animals: the data acquired by the sensors are processed to implement highly complex behaviors, exactly as animals and human beings do.

Figure 1.2. Manipulator performing industrial task for EuRoC project. Retrieved from `www.euroc-project.eu/`.

Each sensor retrieves heterogeneous measurement either of the internal conditions in which the robot is operating (*proprioceptive* sensors), or of the external surrounding environment (*exteroceptive* sensors). Within this perspective, vision is typically accomplished through camera sensors of different types. Indeed, usage of camera sensors is widely and transversely considered as part of the standard equipment, among different deployed platforms of robot applications.

Industrial manipulator arms equip camera sensors to solve complex manufacturing and logistic tasks, e.g., welding, assembly, packaging, where the object to be manipulated needs to be also identified. Interactive manufacturing has been one of the three industry-relevant challenges proposed by European Competition EuRoC[2], born with the goal of boosting advanced research-confined technologies on the industrial field (Figure 1.2). Logistics and inspection were the other two areas addressed by EuRoC, involving use of mobile and aerial robots. Particularly, unmanned aerial vehicles (UAV) are receiving a great boost both in research activities and in industrial and commercial fields. UAVs, in fact, have revealed to be extremely helpful and effective in addressing exploration, inspection and rescuing tasks (Figure 1.3a)), in particular in outdoor environments regarding unaccessible areas, e.g., due to natural calamities or nuclear disasters. Nevertheless, the problem of solving articulated and complex tasks in human-unaccessible environments has been the great motivation of the DARPA Robotics Challenge[3], that put the focus on human-tailored tasks addressed by humanoid robots (Figure 1.3b). Indeed, humanoids represent a further huge class of robotic platforms that is actually facing an increasing interest also outside the walls of laboratories and research centers, due to their highly redundant structure and the possibility to control the whole body. Therefore, specific tasks, such as object manipulation and navigation in complex environments, are achievable by inspiring and replicating human actions and behaviors. Also in this case, the usage of vision is fundamental: in all the tasks involved during the DARPA Challenge, the information acquired by equipped cameras was crucial to identify manipulating objects and estimate their distances, as well as detect safe areas of uneven terrains on which steps could be placed.

This holds also for robots employed for the RoboCup, where a team of humanoids NAO compete in soccer matches (Figure 1.3c): the ball, opponents and

[2] `http://www.euroc-project.eu/`
[3] `https://www.darpa.mil/program/darpa-robotics-challenge`

(a) (b) (c)

Figure 1.3. (a) Unmanned aerial vehicle (UAV) inspecting a tree with a camera sensor. Retrieved from [5]. (b) Humanoid robot driving a vehicle during the DARPA Robotics Challenge. (c) Humanoid robot NAO of the SPQR team involved in the Robocup.

the traversable space on the field are detected by processing images acquired by the monocular cameras mounted on the robot heads.

In surgical robotics, visual information is required to the surgeon to provide an enhancement of his/her visual capabilities and improve the performances of surgical operations. In this particular case, this can be achieved by also considering virtual or augmented reality systems, where the standard visual information acquired by the image can be increased by considering augmented reality displays, showing additional and helpful information to the user observing the scene. Within this context, the da Vinci robotic system is the most recent and widely-employed example. The 3D vision system of the master console enables the surgeon to have a close view of the operating scene, performed by the tele-operated slave manipulators of the system, in a comfortable way (Figure 1.4).

We want to focus on these two latter classes of robots, i.e., mobile humanoid and surgical robots. Therefore, in this manuscript we propose some vision-based methods to solve control and state estimation problems for mobile and surgical robotics. For humanoid robots, moving within an environment highly populated by obstacles can be a hard task to achieve, since the actual traversable and safe space can be reduced. This is a widely addressed problem in literature, solved with different paradigms that could consider a map building process (e.g., SLAM, path planning) or reactive strategies to safely navigate among the obstacles. However, there are not formal results of convergence that can guarantee safety of the robot in particular benchmark test cases. Therefore, we propose a purely reactive safe navigation scheme for wheeled and humanoid robots, moving in unknown environments populated by obstacles, and we show that is possible to formally guarantee safety preservation. For this task, dense visual information is used to perceive the environment (i.e., detect ground plane and obstacles) and, in combination with other sensory sources, provide an estimation of the robot motion with a linear observer. On the other hand, sparse visual data are extrapolated in terms of geometric primitives, in order to implement a visual servoing control scheme satisfying proper navigation behaviors. This controller relies on visual estimated information and is designed in order to guarantee safety during navigation. In addition, redundant structures are taken into account to re-arrange the internal configuration of the robot and reduce its encumbrance when the workspace is highly cluttered.

Second, the focus is moved on surgical robots, specifically on the da Vinci robotic

system. In the field of surgical robotics, having reliable data about unmeasurable quantities is of great importance and critical at the same time. This is particularly true for semi-autonomous robot-assisted surgical procedures, where knowing the state of the tools currently handled by the platform (e.g., position and orientation of a needle) is critical for the success of the operation. Most of efforts in research relies on neural-networks-based approaches, due to the highly unstructured property of the surgical environment, that makes the process of image-based identification really hard. However, the possibility to tackle this problem by considering dynamic observers has not been largely explored yet. Therefore, in this manuscript, we present a preliminary study to estimate the 3D pose of a suturing needle held by a surgical manipulator for robot-assisted suturing, by using a Kalman Filter observer. This is a relevant problem, since it is important for the surgeon has reliable information to accomplish tiring tasks such as wound stitching. The method exploits images acquired by the endoscope of the robot platform to extrapolate relevant geometrical information and get projected measurements of the tool pose. This method has also been validated with a novel simulator designed for the da Vinci robotic platform, with the purpose to ease interfacing and employment in ideal conditions for testing and validation.

The manuscript is concluded by presenting a preliminary study about the *active sensing* paradigm, where the inputs of a system of interest are generated to accomplish some optimality criteria on the estimation performances. The Kalman-based observers mentioned above are classical *passive* estimators, whose system inputs used to produce the proper estimation are theoretically arbitrary. This does not provide any possibility to *actively* adapt input trajectories in order to *optimize* specific requirements on the performance of the estimation. For this purpose, *active estimation* paradigm is introduced and some related strategies are presented. More specifically, a novel active sensing algorithm employing visual dense information is described for a typical Structure-from-Motion (SfM) problem. The algorithm generates an optimal estimation of a scene observed by a moving camera, while minimizing the maximum uncertainty of the estimation. This approach can be applied to any robotic platforms and has been validated with a free-flying monocular camera.

Figure 1.4. Surgeon operating with the da Vinci robotic system observes the clinical scene with the 3D vision system of the master console.

In detail, the manuscript is organized as follows: Chapter 2 present the vision-based navigation problem for wheeled and humanoid robots. Chapter 3 presents the da Vinci robotic system, along with the developed simulating tools and the needle pose estimation scheme. Chapter 4 presents the preliminary study about *active sensing* paradigm in control systems. Chapter 5 concludes the manuscript.

Chapter 2

Vision-based navigation for omnidirectional robots

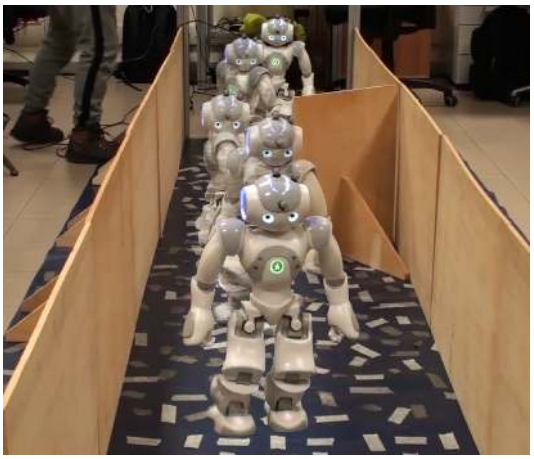


Figure 2.1. Example of a challenging scenario where the robot can exploit both omnidirectional mobility and body re-orientation.

Mobile robots, by construction, have an infinite workspace inside which they need to move in a safe way to accomplish tasks in industrial and service contexts. The potentially unlimited size of their workspace, and the likelihood of changes prevent or make inefficient to structure the environment geometry and sensory equipment.

Examples include robots employed, e.g., in logistics or for assistive tasks in office-like environments, typically cluttered and dynamically changing. This is also the case of robots employed for exploration or rescuing tasks in post-disaster environments, where critical and precarious conditions of the location make the workspace highly unstructured and uncertain for the robot, and the deployment of external sensory services inefficient or impossible.

The examples provided above refer to canonical robot navigation problem, that is an instance of a more general motion planning problem: the robot is demanded to move within an environment, that can be unknown and reconstructed on-line (*on-line* planning), or assumed as known in advance (*off-line* planning), in order to satisfy some particular assigned behavior or reach a predefined goal. The environment

can be populated by obstacles, thus the robot needs to plan an obstacle-free path (*global* approaches)[6], or an instantaneous motion (*local* approaches)[7], to guarantee safety. To manage this task, robots exploit information acquired by available sensors, that can be on-board (encoders, IMU, cameras) or integrated in the environment (external tracking system, e.g., VICON). In this sense, exteroceptive sensors like cameras are fundamental to reconstruct reliable geometric information about the surrounding areas and are crucial to detect and identify possible obstacles.

Furthermore, one typically takes into account also instantaneous motion limitations due to non-holonomic constraints of the robot platform, to generate feasible paths that can be actually executed by the robot. However, these constraints may reduce the space of possible configurations and trajectories that can bring the robot to the goal, or to satisfy its task. On the other hand, omnidirectional robots, both wheeled and legged, offer a higher degree of manoeuvrability that can ease navigation in challenging scenarios, like moving sideways in narrow passages (see Fig. 2.1). In conjunction with a maneuverable platform, navigation in an unstructured, unknown environment requires also to endow the robot with reactive capabilities allowing to complete the task while avoiding unsafe contacts with the surrounding environment.

In this chapter, we focus on vision-based navigation problems, where visual information is necessary for the accomplishment of the task. Specifically, we consider a local strategy where the robot reactively moves based on the instantaneous information acquired by sensors. From this point of view, vision-based control methods like image based visual servoing (IBVS) [8] are particularly appealing due to the possibility to define the task in the sensor space, without the need of a working space map nor of the robot pose within the environment. In addition, monocular cameras are commonly available also on cheap robotic platforms.

Despite the success of humanoid robots in recent years and the quite common use of omnidirectional wheeled robots in the industrial context, not so many work address the problem of omnidirectional robot navigation in environment populated by obstacles [9][10][11]. Indeed, navigation problem for omnidirectional wheeled robots is not extensively addressed. In [9], an obstacle avoidance navigation scheme is proposed, where a number of drivable paths are defined to generate the omnidirectional motion and avoid obstacles, while reaching a visible target. However, the presented method uses laser scans to perceive the surrounding environments and does not exploit visual information. Moreover, the pre-specified paths can limit the full potential of an omnidirectional motion.In [10], a full omnidirectional navigation system is presented. The proposed scheme solves a simultaneous mapping and localization (SLAM) problem, while generating an optimized trajectory that exploits the efficient manoeuvrability of the omnidirectional platform. In the proposed scheme, the motion is planned in advance, as no reactive behaviours are taken into account, and visual information is not used.

On the other hand, the vision-based navigation problem for humanoid robots has been widely addressed. A comprehensive description of the general problem, highlighting related SLAM and planning tools, is given in [12]. In [13], an energy-efficient approach for indoor environment exploration is presented. The method generates the view poses for the inspection based on several constraints, in order to cover the whole 3D scene. The environment is assumed to be known, thus no reactive behaviors are designed. In [14], a vision-based navigation framework, with

obstacles detection, is achieved for a humanoid robot in a learning fashion. The robot detects and classifies the obstacles in order to identify the traversable space of the robot. However, the detection requires also the use of laser scan data, along with the information acquired from monocular camera images. In [15], a vision-based navigation scheme has been presented for a humanoid robot in a maze-like environment. The robot processes the images from the camera to extract a pair of visual guidelines that identifies the navigating corridor, and use an IBVS control scheme to regulate the robot at the center of the corridor. The scheme considers the robot moving according to a unicycle motion model, and assumes that the corridor is large enough to allow safe navigation and is obstacle-free.

In [16], the problem of navigation for a humanoid robot is addressed through the use of a planner based on a pre-built visual memory. The planner takes into account the presence of unexpected obstacles and exploits the omnidirectional motion capabilities of the humanoid robot to implement obstacle avoidance behaviors. However, since the robot is not able to look towards the direction in which it is moving, there is no guarantee that other unseen obstacles are successfully avoided.

In the works cited above, the presence of actuation errors in the robot model has not been considered. This is a relevant aspect since, when system inputs are not properly actuated (e.g., because of modeling errors, mechanical faults or external disturbances), the effectiveness of the control can be compromised and safety not preserved. For this reason, in the following we show different proposed solution, addressing the problem of safe navigation for omnidirectional robot, both in the nominal case, i.e., when the robotic system inputs are correctly actuated, and in the perturbed case, i.e., when the presence of actuation disturbances is taken into account. Specifically, this issue is addressed in two different ways, based on information extracted by the available sensors. In one case, the disturbance is rejected through the augmentation of the considered visual task, by defining additional visual features to be regulated in the image. In the other case, the robot velocity is estimated through a Kalman Filter combining optical flow-based ego-motion visual measurements with inertial and odometry data extracted from IMU and joint encoders. Then, a velocity loop is closed to compensate the discrepancy between actual and nominal inputs.

A part of the discussion described below, concerning navigation of humanoid robot in nominal conditions, can be found in [17]. At the time of the submission of this manuscript, the remaining contents, regarding the generalization to mobile omnidirectional robots and the handling of actuation disturbances, are about to be submitted on a journal paper[18].

2.1 Problem description and proposed approach

We address the problem of safe navigation for omnidirectional mobile robots, within an unknown environment cluttered by obstacles. In this context, we assume that the robot relies on its on-board sensory equipment only, while nothing is assumed about the geometry of the environment, except for the local flatness of the ground. Within the on-board sensory equipment, we consider a monocular camera mounted on the robot platform. The camera is used to perceive the traversable space and possible

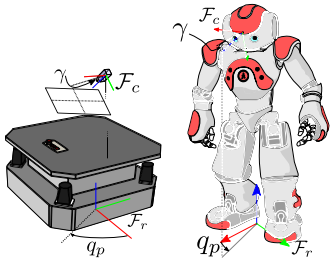


Figure 2.2. Omnidirectional wheeled (a) and humanoid (b) robot with reference frames of interest. We also show tilt angle γ and the pan joint angle q_p.

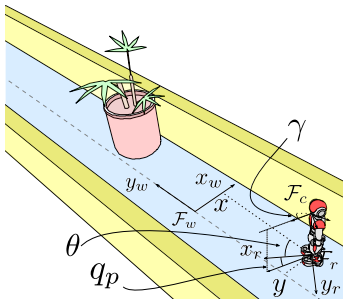


Figure 2.3. The pose of the robot in the world frame $\mathcal{F}_w$ is denoted by the position (x, y) and the orientation θ.

obstacles by extracting proper information from acquired images. In addition, we want to take into account the possibility that the system inputs are not properly actuated, e.g., due to modeling errors or mechanical faults. With reference to Fig. 2.2 and Fig. 2.3, we consider the following frames of interest: (i) the world inertial frame $\mathcal{F}_w$; (ii) the camera frame $\mathcal{F}_c$, with z-axis along the camera principal axis and defined according to the "right-bottom" convention; (iii) the robot frame $\mathcal{F}_r$, with the x- and y-axis directed along forward and left directions of the platform, respectively. The orientation of the camera with respect to $\mathcal{F}_r$ is expressed through pan and tilt angles, denoted by q_p and γ, respectively. The projection on the ground of the camera position in $\mathcal{F}_w$ is given by the pair of coordinates (x, y). Denoting by θ the absolute orientation of the robot body with respect to y_w, the absolute camera orientation is given by $\theta_c = \theta + q_p$. Assuming a planar motion, the robot commands for navigation, expressed in $\mathcal{F}_r$, are: (i) the Cartesian linear velocities v_x and v_y; (ii) the angular velocity ω_z, denoting the change of orientation of the robot body. Finally, we assume that the robot can also actuate the camera pan angle q_p through a joint velocity u_p, so as to control the camera orientation independently from that of the body.

Given these working conditions, the most direct approach would be to steer the robot toward regions of maximum clearance from image obstacles, while moving in the direction of the gaze of the camera and changing the internal configuration of the robot in presence of narrow passages generated by obstacles. Therefore, we first proposed an approach that allows the navigation of omnidirectional robots in a

priori unknown environments, populated by obstacles of any shape, possibly with narrow passages, that consists in the following aspects:

- enforce a motion model that aligns the direction of motion with the gaze of the camera;
- visual servoing techniques to regulate suitable visual features, processed in the camera images, to a desired value which maximizes the averaged distance of obstacles images to the image plane ordinate axis;
- internal robot reconfiguration to favor narrow passage crossing.

These behaviors are achieved in the following way: to allow proper reconfiguration of the robot, we consider a decoupled control of the camera pan angle q_p with respect to the control of the robot body orientation. This is combined with the explicit enforcement of a constraint imposed on v_x and v_y, to change and align the instantaneous direction of motion of the platform with the (varying) camera principal axis. As a result, the robot/camera system always moves in the direction of its optical axis.

We observe that maximizing the distance of obstacles to the image center, in general, does not imply that the robot is moving with the maximum clearance to all the workspace obstacles. However, in case of navigation along a corridor, the constraint on the camera velocity allows to formally prove convergence to the corridor bisector, similarly to [19]. With respect to [19], the approach proposed here does not rely on geometric assumptions about the environment and it allows for the presence of obstacles of any shape and uses omnidirectional walk to cross narrow passages. In the following, we propose a general navigation scheme for the considered problem, as it has been presented above.

Specifically, the scheme considers a preliminary image processing procedure to generate the proper visual features required for the control strategy. Then, the described navigation behaviors are achieved by proposing different vision-based control strategies. We evaluate and compare the effectiveness of such strategies both when the system correctly actuates the control inputs (nominal case), and when the presence of actuation disturbances in the robotic system is taken into account (perturbed case). As will be seen, in both cases, to guarantee safety for the robot any time during navigation, it is important to ensure that motion direction and the the gaze of the camera are kept aligned. This requirement can be disregarded during the transient response of some visual controllers, thus compromising safety of the robot during navigation.

For these reasons, we finally propose a further controller, that aims to directly compensate the error between the actual velocity and the desired velocity, in order to cope with the scenarios when the alignment constraint is not ensured. The actual robot velocity is obtained through a Kalman Filter that fuses a vision-based velocity measurement, recovered from dense optical flow, with inertial and proprioceptive measurements, coming from IMU and joint encoders.

In the next Section, we present the navigation algorithm realizing omnidirectional navigation in the considered conditions.

(a) Optical flow. **(b)** Planar flow. **(c)** Dominant plane.

Figure 2.4. Main steps of the image processing algorithm.

2.2 Omnidirectional Navigation Algorithm

To satisfy the proposed objectives, the proposed algorithm consists in two main modules: (i) a perception module, where a preliminary image processing procedure detects obstacles in the image and extracts the suitable visual features for the tasks; (ii) a vision-based control module, that uses the visual information extracted in the perception module to accomplish a properly defined task. We will detail this procedure in the next paragraph.

2.2.1 Perception

The images acquired by the monocular camera are used to generate the proper visual features required to the control strategy. For this purpose, we adopted a RANSAC-based algorithm for the detection of the *dominant plane*, meant as the appearance of the traversable space in the camera image. The method uses a measure of the *dense* optical flow evaluated between a pair of consecutive images, describing the apparent motion of the workspace on the image plane. In the following, we briefly summarize this algorithm, that is fully described in [20].

At each time instant, the algorithm first computes the dense optical flow image, to have a measurement of the apparent motion of each image pixels. Computing the optical flow is equivalent to solve the following equation:

$$\frac{\partial I(x,y,t)}{\partial x}\dot{x} + \frac{\partial I(x,y,t)}{\partial y}\dot{y} + \frac{\partial I(x,y,t)}{\partial t} = 0 \tag{2.1}$$

where $\boldsymbol{x} = (x,y)^T$ is a generic pair of coordinates of the gray-scale image $I(x,y,t)$ at time t. The equation (2.1) is a well-known ill-posed problem, so optical flow solving methods typically adopt additional constraints to generate a consistent measurement of the optical flow [21] [22] [23]. Whatever implementation is chosen, the resulting optical flow image corresponds to a vector field $\boldsymbol{u}(x,y,t) = (\dot{x},\dot{y})$ of velocity vectors.

Then, by observing that points on the ground plane move differently in the image with respect to points related to other objects of the scene, the computed optical flow is used to estimate the apparent motion of the ground, that is referred as *planar flow*. So, comparing the measured optical flow with the estimated planar flow allows to distinguish pixels lying on the ground from pixels belonging to other objects. Such planar flow is built by robustly estimating the 3×3 homography matrix $\boldsymbol{H}$ induced by the plane between two consecutive camera images, in a RANSAC fashion.

Specifically, under the assumption that the camera motion between two subsequent camera frames is small, $\boldsymbol{H}$ can be approximated to an affinity transformation applied on the point correspondences $\boldsymbol{x} \leftrightarrow \boldsymbol{x'}$:

$$\boldsymbol{x\prime} = \boldsymbol{H}\boldsymbol{x} \approx \begin{bmatrix} \boldsymbol{A} & \boldsymbol{b} \\ \boldsymbol{0}^T & 1 \end{bmatrix} \boldsymbol{x} \tag{2.2}$$

where $\boldsymbol{x} = (x, y, 1)^T$ and $\boldsymbol{x\prime} = (x\prime, y\prime, 1)^T$ are the vectors of homogeneous coordinates of $\boldsymbol{x}$ and $\boldsymbol{x\prime}$ respectively, $\boldsymbol{A}$ is a 2×2 matrix and $\boldsymbol{b}$ is a two-dimensional vector. The coefficients $\boldsymbol{A}$ and $\boldsymbol{b}$ are robustly estimated with the following RANSAC-based method, for each pair of subsequent frames $I(x, y, t)$ and $I(x, y, t + 1)$:

1. a minimal set of correspondences $\boldsymbol{x} \leftrightarrow \boldsymbol{x\prime}$ is randomly chosen among the optical flow field $\mathbf{u}(x, y, t)$. Since we need to estimate 6 parameters, three pairs of points are required:

$$\boldsymbol{x}_i\boldsymbol{\prime} = \boldsymbol{x}_i + \boldsymbol{u}(x_i, y_i, t), \quad i = 1, 2, 3 \tag{2.3}$$

2. A model $\left(\hat{\boldsymbol{A}}, \hat{\boldsymbol{b}}\right)$ is instantiated from the three random correspondences $\boldsymbol{x} \leftrightarrow \boldsymbol{x\prime}$, by arranging and solving (2.2) in the form

$$\mathbf{M}\boldsymbol{h} = \mathbf{0} \tag{2.4}$$

   where $\boldsymbol{M}$ is a matrix of proper coefficients, and $\boldsymbol{h} = (a_{11}, a_{12}, a_{21}, a_{22}, b_1, b_2)^T$ is the vector of unknown entries of $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{b}}$.

3. For each point $\boldsymbol{x}$ in the image, the estimated *planar flow* vector is computed:

$$\hat{\boldsymbol{u}}(x, y, t) = \hat{\boldsymbol{x}}\boldsymbol{\prime} - \boldsymbol{x} = \hat{\boldsymbol{A}}\boldsymbol{x} + \hat{\boldsymbol{b}} - \boldsymbol{x} \tag{2.5}$$

   and determine the *consensus set*, i.e. the set of points such that

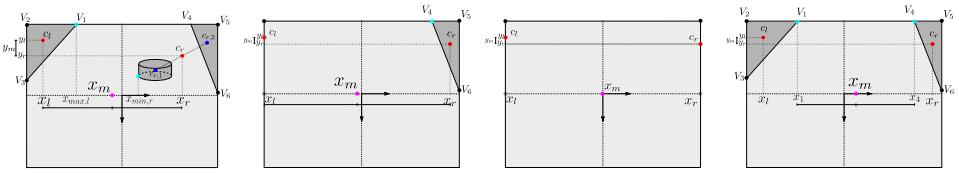
$$\|\boldsymbol{u}(x, y, t) - \hat{\boldsymbol{u}}(x, y, t)\| < \epsilon \tag{2.6}$$

4. the *dominant plane* image $\boldsymbol{d}(x, y, t)$ is built as

$$\boldsymbol{d}(x, y, t) = \begin{cases} 255, & \text{if } \|\boldsymbol{u}(x, y, t) - \hat{\boldsymbol{u}}(x, y, t)\| < \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{2.7}$$

5. If the size of the consensus set (*inliers*) is greater than a given threshold T, the coefficients $(\boldsymbol{A}, \boldsymbol{b})$ are robustly re-estimated using all the detected inliers, and the algorithm terminates. Otherwise, a new set of three correspondences $\boldsymbol{x} \leftrightarrow \boldsymbol{x\prime}$ is randomly selected again and the procedure is repeated. If, after N trials, the threshold is not reached, the largest consensus set is used to get the final robust estimation of $\boldsymbol{A}$ and $\boldsymbol{b}$.

Erosion and dilation operators are further applied to $\boldsymbol{d}(x, y, t)$, to remove salt-and-pepper noise and obtain the final image in Fig. 2.4c. At the end of the process, white pixels in $\boldsymbol{d}(x, y, t$ represent the ground as the traversable space for the robot,



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

Figure 2.5. Visual features recurring in: (a) a general case of environment cluttered by obstacles; (b) an ill-conditioned case where one of the obstacle is missing and the centroid is placed on the image border; (c) ill-conditioned case where both centroids are missing; (d) a simplified scenario with a clear corridor.

while black pixels represent the obstacle region. This information is used to compute the visual features required for the proposed control strategy.

Specifically, each connected area of image pixels represents an obstacle $\mathcal{O}_i$. For each $\mathcal{O}_i$, we compute the corresponding centroid in normalized camera coordinates $\boldsymbol{c}_{i,s} = (x_{i,s}, y_{i,s})$ ($s = \{l, r\}$ depending on the considered side of the image):

$$\boldsymbol{c}_{i,s} = \frac{1}{N_{i,s}} \sum_{j=1}^{N_{i,s}} \boldsymbol{x}_j \tag{2.8}$$

where $N_{i,s}$ is the area of $\mathcal{O}_i$. Then, we generate the point features $\boldsymbol{c}_l = (x_l, y_l)$ and $\boldsymbol{c}_r = (x_r, y_r)$

$$\begin{aligned} \boldsymbol{c}_l &= \tfrac{1}{n_l} \textstyle\sum_{i=1}^{n_l} \boldsymbol{c}_{i,l} \\ \boldsymbol{c}_r &= \tfrac{1}{n_r} \textstyle\sum_{i=1}^{n_r} \boldsymbol{c}_{i,r} \end{aligned} \tag{2.9}$$

as the average of all the n_l and n_r centroids $\boldsymbol{c}_{i,l}$ and $\boldsymbol{c}_{i,r}$ on the left and right side of the image, respectively. Finally, we define the following visual features (Fig. 2.5)

$$\begin{aligned} x_m &= \frac{1}{2}(x_l + x_r) \\ y_m &= (y_l - y_r) \end{aligned} \tag{2.10}$$

where x_m denotes the abscissa of the middle point between $\boldsymbol{c}_l$ and $\boldsymbol{c}_r$, while y_m is the difference of ordinates. As will be explained in the next Section, the choice of these visual features is motivated by the fact that regulating both x_m and y_m to 0 is equivalent to maximize the distances of the robot from the workspace obstacles. When navigating in a corridor, this corresponds to navigating along the corridor bisector.

It is worth noting that, in principle, the definitions (2.10) are well-defined if at least one obstacle is visible on both side of the image. However, this may be not true in general, since obstacles may appear only in one (or either) side of the image. In these cases, x_m and y_m are ill-conditioned. To address these scenarios, the position of a missing centroid is fictitiously assigned on the image border, with ordinate equal to the latest available value (see Fig. 2.5b-2.5c). Obviously, this expedient will alter the motion of x_m and y_m on the image. As will be clear in next Section, while this artifact has not a strong influence on the dynamics of x_m, it may significantly affect

the dynamics of y_m, and the satisfaction of the control objectives considered for the task may not be guaranteed.

Finally, we evaluate the minimal image distance $d = |x_{max,l} - x_{min,r}|$ between the contours of right and left obstacle areas, where $x_{max,l}$ (resp., $x_{min,r}$) is the abscissa of the rightmost (resp., leftmost) pixel in the set of contours of left (resp., right) obstacles. This value is assumed as a measure of the passage width, and we use it to generate a reference value q_p^* for the control of the camera pan angle according to the following criterium:

$$q_p^* = \begin{cases} q_{p,max}, & \text{if } d < t_l \\ 0, & \text{if } d > t_h \\ q_{p,max} - (\frac{d-t_l}{t_h-t_l}) * q_{p,max}, & \text{if } t_l < d < t_h \end{cases} \tag{2.11}$$

where $q_{p,max}$ is the maximum pan joint position allowed, and $t_l = \left\{t_l^-, t_l^+\right\}$ and $t_h = \left\{t_h^-, t_h^+\right\}$, experimentally found, are evaluated through an hysteresis thresholding, to get a measurement of $d_{x,min}$ robust to the image noise, and such that $t_l < t_h$. This definition has the following meaning: when the measured distance d is sufficiently high (i.e., when the passage width is large enough), the orientation of the camera is not modified and the robot moves in the direction of the torso orientation. When d goes below the threshold d_{max} (i.e., when the passage width is reducing), q_p^* varies linearly from 0 to $q_{p,max}$, value reached for $d \leq d_{min}$. When the camera pan angle q_p successfully reaches q_p^* and the direction of motion is aligned with the camera principal axis through (2.12), the robot moves sideways, minimizing its encumbrance in the workspace and risk of collision. It is worth to highlight that in principle the 3D width of the available free space could be computed from geometric and projective reasoning, using the information that is already available. This would allow to precisely evaluate the thresholds required for the computation of q_p^*. This is an aspect that will be investigated in the future.

2.2.2 Vision-based control

The desired navigation behaviors described in the previous section can be achieved by considering the following control objectives: (i) enforce the alignment between the direction of motion and the gaze of the camera; (ii) maximize the clearance from obstacles on the image plane; (iii) control the camera pan angle based on the amount of perceived traversable space.

The control objective (i) is easily formalized by requiring that

$$\begin{cases} v_x = v\cos(q_p) \\ v_y = v\sin(q_p) \end{cases} \tag{2.12}$$

where $v = \sqrt{v_x^2 + v_y^2}$ is the norm of the linear velocity vector. Satisfying this constraint is crucial for the success of the navigation task. Therefore, for the control strategies that will be presented in this section, we consider as performance evaluation metrics the alignment error between the camera principal axis and the robot motion direction. Formally, denoting by $\hat{\boldsymbol{v}} = 1/v[v_x, v_y]^T$ the motion direction

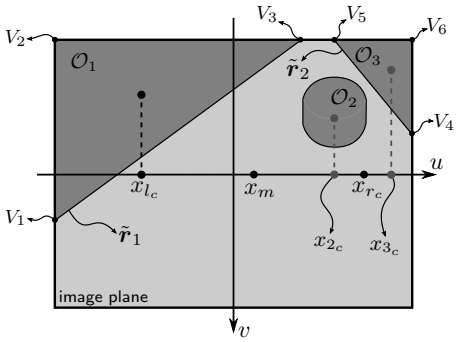


Figure 2.6. Geometric entities recurring in the proof of Proposition.

and by $\hat{\boldsymbol{z}}_g = [\cos q_p, \sin q_p]$ the camera principal axis vector projected on the ground, the alignment error is computed as

$$\phi = \cos^{-1}(\hat{\boldsymbol{v}}^T \hat{\boldsymbol{z}}_g) \tag{2.13}$$

To satisfy the control objectives (ii) and (iii), we exploit the visual information extracted through image processing, as shown in Sect. 2.2.1, i.e., the visual features x_m and y_m and the image obstacle distance d. Therefore, in the reminder of this chapter, we first determine the relationship between the velocity of the visual features x_m and y_m and the robot velocity input available for control, in Sect. 2.2.3. Then, in Sect. 2.2.4, we show different control schemes that, using this visual information, steers the robot among the environment obstacles while enforcing the motion model to align with the camera gaze, and reconfiguring to favor narrow passage crossing. In particular, the proposed controllers guarantee exponential convergence of the camera frame to the center of a corridor with the optical axis aligned with the corridor bisector. Based on the size of the traversable space, the robot body is oriented so as to reduce its transversal encumbrance. This behavior is obtained by exploiting the humanoid omnidirectional walk allowing robot velocity directions outside its sagittal plane.

2.2.3 Dynamics of the visual features

For each obstacle $\mathcal{O}_i$ detected on the image, the dynamics of the corresponding centroid $\boldsymbol{c}_{i,s}$ is related to the 6-D camera velocity $\boldsymbol{v}_c$ by [24]:

$$\dot{\boldsymbol{c}}_{i,s} = \boldsymbol{L}_{i,s} \boldsymbol{v}_c \tag{2.14}$$

where

$$\boldsymbol{L}_{i,s} = \begin{bmatrix} \boldsymbol{L}^x_{i,s} \\ \boldsymbol{L}^y_{i,s} \end{bmatrix} = \frac{1}{N_{i,s}} \sum_{j=1}^{N_{i,s}} \begin{bmatrix} -\frac{1}{Z_j}, & 0, & \frac{x_j}{Z_j}, & x_j y_j, & -1 - x_j^2, & y_j \\ 0, & -\frac{1}{Z_j}, & \frac{y_j}{Z_j}, & 1 + y_j^2, & -x_j y_j, & -x_j \end{bmatrix} \tag{2.15}$$

is the average of the interaction matrices related to the $N_{i,s}$ pixels (x_j, y_j), with depth Z_j, that constitute $\mathcal{O}_i$.

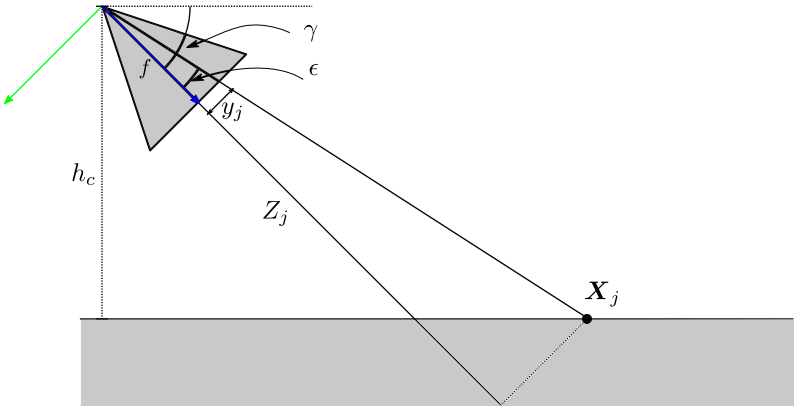


Figure 2.7. Geometric reconstruction of depth of points lying on the ground.

The reader observes that, given the definition of centroid (2.8), the corresponding dynamics as given in (2.15) assumes that the area $N_{i,s}$ is constant, i.e., the appearance of the obstacle occupies always the same number of pixels in the image. Clearly, this is not always true, as the area of an object projected on the image plane changes as the camera is approaching. In addition, the limited field-of-view also alters the areas of the obstacles, when they enter or exit from the image plane, in a way that is not predictable if we do not make any *a priori* assumption about the geometry of the environment. However, when the dynamics of both camera and environment are slow with respect to the control rate (as in this case), this approximation will not affect the system performance.

Furthermore, since images are acquired from a monocular camera, we do not know the exact depth Z_j of each point, but we can reconstruct geometrically the depth of the points lying on the ground. Specifically, denoting by h_c and γ the height and the tilt angle of the camera, respectively, we can reconstruct geometrically the depth Z_j of a point $\boldsymbol{X}_j$ lying on the ground as

$$Z_j = \frac{h_c \cos\epsilon}{\sin(\gamma + \epsilon)} \tag{2.16}$$

where $\epsilon = \text{atan2}(y_j, f)$, being y_j the pixel ordinate of the point $\boldsymbol{X}_j$ projected on the image plane, and f is the focal length of the camera. Therefore, to evaluate 2.15, we consider an approximation of Z_j by projecting the corresponding point $\boldsymbol{x}_j$ on the ground, thus over-estimating the real value of the depth. Visual servoing schemes are typically robust to these types of approximations, as explained in [8]. In the experimental section, we will show that these approximations provides satisfactory results. We also highlight that the geometric concept of centroid comes from the discrete definition of geometric image moments. Since we are assuming that the obstacle points lie on the ground, one could also use the dense definition of moments in a more elegant way, as only the three parameters of the ground plane would have been involved. This alternative has not been validated but it will be investigated in the future.

From (2.10) and (2.15), we derive the dynamics of x_m and y_m as

$$\begin{bmatrix} \dot{x}_m \\ \dot{y}_y \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left(\frac{1}{n_l}\sum_{i_l=1}^{n_l} \boldsymbol{L}_{i_l}^x + \frac{1}{n_r}\sum_{i_r=1}^{n_r} \boldsymbol{L}_{i_r}^x\right) \\ \frac{1}{n_l}\sum_{i_l=1}^{n_l} \boldsymbol{L}_{i_l}^y - \frac{1}{n_r}\sum_{i_r=1}^{n_r} \boldsymbol{L}_{i_r}^y \end{bmatrix} \boldsymbol{v}_c = \begin{bmatrix} \boldsymbol{L}_{x_m} \\ \boldsymbol{L}_{y_m} \end{bmatrix} \boldsymbol{v}_c = \boldsymbol{L}\boldsymbol{v}_c \tag{2.17}$$

The dynamics of the visual features can be related to the robot 6-D velocity $\boldsymbol{v}_r$ via

$$\begin{bmatrix} \dot{x}_m \\ \dot{y}_m \end{bmatrix} = \boldsymbol{L}\ {}^c\boldsymbol{W}_r\ \boldsymbol{v}_r = \boldsymbol{J}_m \boldsymbol{v}_r, \tag{2.18}$$

with ${}^c\boldsymbol{W}_r$ the twist velocity transformation matrix:

$${}^c\boldsymbol{W}_r = \begin{pmatrix} {}^c\boldsymbol{R}_r & {}^c\boldsymbol{S}_r\,{}^c\boldsymbol{R}_r \\ 0 & {}^c\boldsymbol{R}_r \end{pmatrix}, \tag{2.19}$$

${}^c\boldsymbol{R}_r$ the rotation matrix expressing the robot frame with respect to the camera frame, and ${}^c\boldsymbol{S}_r$ the skew matrix associated to the translation of the robot frame with respect to the camera frame. Both ${}^c\boldsymbol{R}_r$ and ${}^c\boldsymbol{S}_r$ can be deduced from the robot proprioception.

Finally, assuming planar motion, and taking into account the contribution of the pan joint velocity $\dot{q}_p$, the relationship between the dynamics of the features x_m and y_m and the desired control inputs is given by

$$\begin{bmatrix} \dot{x}_m \\ \dot{y}_m \end{bmatrix} = \boldsymbol{J}\boldsymbol{v}_r = \begin{bmatrix} J_{v_x}^x & J_{v_y}^x & J_{\omega_z}^x & J_{\omega_z}^x \\ J_{v_x}^y & J_{v_y}^y & J_{\omega_z}^y & J_{\omega_z}^y \end{bmatrix} \begin{bmatrix} \mathbf{v}_r \\ u_p \end{bmatrix} = \boldsymbol{J}\mathbf{v}_r + \boldsymbol{J}_{\omega_z} u_p \tag{2.20}$$

being $\mathbf{v}_r = [v_x\ v_y\ \omega_z]$ and $\boldsymbol{J}_{\omega_z}$ the third column of the 3×2 submatrix $\boldsymbol{J}$ of $\boldsymbol{J}_m$. Note that the effect of $\dot{q}_p$ on the feature velocity is identical to that of ω_z under the assumption that camera pan and robot rotation axes are aligned.

Given the dynamics (2.20), we propose different visual control strategies to solve the described navigation problem, as detailed in the next paragraph.

2.2.4 Control design

Our objective is to define a controller that orients the robot body along the direction of maximum clearance to the environment obstacles, while considering internal reconfiguration of the robot structure to address narrow passage crossings.

This behavior is obtained by regulating a) a properly defined visual error to zero, and b) the head pan joint q_p to a desired value q_p^*. In defining the visual error a reasonable choice would be to keep the obstacles symmetrical with respect to the vertical axis of the image plane. We can achieve this by acting on the visual feature x_m. In particular, we showed in [17] the relationship between the regulation of x_m to zero and the regulation of the robot pose, in case of navigation in a straight corridor without obstacles. This is resumed with the following result:

Proposition: *Assuming navigation along a straight corridor without obstacles, the regulation of the visual feature x_m to zero, under the constraint (2.12), implies convergence of the camera position and orientation to the corridor bisector., i.e.:*

$$x_m \to 0 \implies (x, \theta_c)^T \to (0, 0)^T \tag{2.21}$$

Proof: Placing the origin of the world reference frame $\mathcal{F}_w$ at the center of the corridor, the corridor guidelines, i.e., the lines at the intersection between the floor and the walls, are represented through the world frame homogeneous coordinates of two points belonging to each line:

$$\boldsymbol{r}_1 = (-1, 0, -d)^T \text{ and } \boldsymbol{r}_2 = (-1, 0, d)^T$$

where d is the semidistance between the two guidelines. These lines project on the image plane as:

$$\hat{\boldsymbol{r}}_1 = \boldsymbol{P}^{-T}\boldsymbol{r}_1 = (a_1, b_1, c_1)^T, \quad \hat{\boldsymbol{r}}_2 = \boldsymbol{P}^{-T}\boldsymbol{r}_1 = (a_2, b_2, c_2)^T$$

where $\boldsymbol{P}$ is the projection matrix relating 3-D points of the cartesian space with 2-D points of the image plane [25]:

$$\boldsymbol{P} = \begin{pmatrix} f_u \cos\theta_c & -f_u \sin\theta_c & -f_u x \cos\theta_c \\ -f_v \sin\gamma \sin\theta_c & -f_v \cos\theta_c \sin\gamma & f_v(h\cos\gamma + x \sin\gamma \sin\theta_c) \\ \cos\gamma \sin\theta_c & \cos\gamma \cos\theta_c & h \sin\gamma - x \cos\gamma \sin\theta_c \end{pmatrix}. \tag{2.22}$$

with f_u, f_v the focal lengths in pixel along the image plane axes. The ordinate y of camera position has been omitted since it is irrelevant for the considered control problem (i.e., only the distance to the bisector is controlled and not the position along the corridor as the robot goal is to move forward). For details on the computation of $\boldsymbol{P}$, refer to [19].

Under the considered working conditions, the corridor walls appear on the image plane as two triangles (see in Fig. 2.6), with vertexes $\boldsymbol{V}_i$, $i = 1, \ldots, 3$ and $\boldsymbol{V}_j$, $j = 4, \ldots, 6$. Computed x_l and x_r respectively as the averaged abscissa of each of the two triangles, using the equation of the projected lines, we have:

$$x_m = \frac{1}{2}(x_l + x_r) = k_1 \frac{x}{c_{\theta_c}} + k_2 \tan\theta_c, \tag{2.23}$$

with $k_1 = f_u/6h(h_i \cos\gamma/f_v - 2\sin\gamma)$, $k_2 = -f_u/6(h_i \sin\gamma/f_v + 2\cos\gamma)$ constants depending on the camera intrinsic and extrinsic parameters, and on the height of the image plane h_i.

From (2.23) it follows that $x_m = 0$ implies $x = -k_2/k_1 \sin\theta_c$. Of the locus of points described by this last equation only $(x, \theta_c) = (0, 0)$ is a stable equilibrium point. In fact, the visual task $x_m = 0$ is satisfied at all times only if the camera is aligned with the corridor bisector due to the enforced mobility model of the camera that keeps the optical axis always aligned with the direction of motion. In other words, if the robot does not move along the bisector the camera will move toward one of the two corridor walls. This will perturb the symmetric position of the obstacles centroids in the image plane provided that

$$k_2/k_1 > 0 \Rightarrow \tan\gamma > \frac{h_i}{2f_v}, \tag{2.24}$$

i.e., the robot camera is sufficiently tilted toward the floor, a condition satisfied by our operational setup. Equation (2.24) is derived from geometric reasoning. □

Remark: *Despite the proof of the proposition above refers to a corridor scenario (i.e., a straight traversable space between two walls projected as triangles in the image), the notion of "corridor" can be meant as a general definition, referring to the traversable space between any pair of obstacles. In this more general sense, even if we do not formally prove that the robot distances from workspace obstacles are maximized, an effective obstacle avoidance behavior can still be accomplished.*

From the proposition above we assert that, to guarantee safety for the robot in the workspace, we can focus on regulating x_m on the image plane. Therefore, the navigation task is formally formulated via the error vector

$$\boldsymbol{e} = \begin{bmatrix} \boldsymbol{e}_v \\ e_p \end{bmatrix} \tag{2.25}$$

where $\boldsymbol{e}_v$ is some visual error and $e_p = q_p - q_p^*$. The definition of $\boldsymbol{e}_v$ - and so the corresponding control law - depends on the type of IBVS controller designed for the task.

In addition, we require that the proposed controllers consider also the presence of actuation disturbances on the considered system. Specifically, we consider disturbances acting on the linear velocity inputs, i.e., the actual linear velocity executed by the robot is given by

$$\begin{aligned} v_x &= \bar{v}_x + \Delta v_x \\ v_y &= \bar{v}_y + \Delta v_y \end{aligned} \tag{2.26}$$

where $\bar{v}_{\{x,y\}}$ denote the nominal inputs computed by the chosen control strategy, and $\Delta v_{\{x,y\}}$ the disturbances acting on the corresponding components.

In the following, we propose some possible control strategies. In the first two cases, we exploit the Null-Space Projector (NSP) of the defined task.

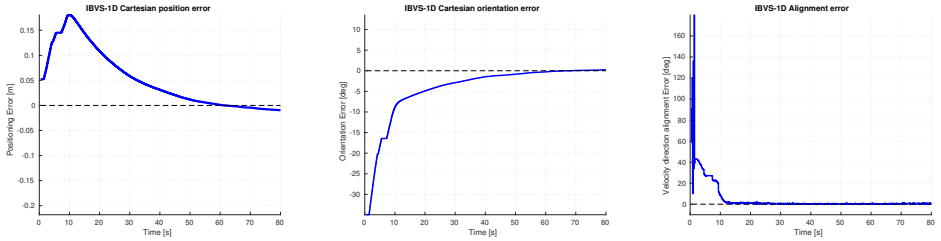
1D IBVS with Null-Space Projection

In this first formulation, we consider the visual task $\boldsymbol{e}_v = e_v = x_m$. Therefore, the dynamics of the error is given by the first row of (2.20), with the task Jacobian given by $\boldsymbol{J}_x$. Specifically, since the matrix $\boldsymbol{J}_x$ has size 1×3, and we have 3 available inputs for the control, the corresponding null-space $\mathcal{N}(\boldsymbol{J}_x)$ has dimension 2. In particular, when $x_m = 0$, a basis of $\mathcal{N}$ is given by

$$\boldsymbol{B}_1 = \left\{ \begin{bmatrix} \cos q_p \\ \sin q_p \\ 0 \end{bmatrix}, \begin{bmatrix} -Zy_r/f \\ 0 \\ \sin q_p \end{bmatrix} \right\}, \tag{2.27}$$

from which we observe that the first vector is equivalent to the unit vector describing the direction of the principal axis of the camera in $\mathcal{F}_r$. In this perspective, the following control law with projection in the null-space

$$\begin{aligned} \mathrm{v}_r &= -\boldsymbol{J}_x^{\#} \left(K_{x_m} x_m + J_{\omega_z}^x u_p \right) + (\boldsymbol{I}_3 - \boldsymbol{J}_x^{\#} \boldsymbol{J}_x) \boldsymbol{v}^* \\ u_p &= -K_p e_p \end{aligned} \tag{2.28}$$



(a) **(b)** **(c)**

Figure 2.8. Simulation results for an omnidirectional wheeled robot navigating in a straight corridor under the 1D-IVBS + NSP control, in nominal conditions. (a) Position Error x. (b) Orientation error θ_c. (c) Alignment error ϕ.

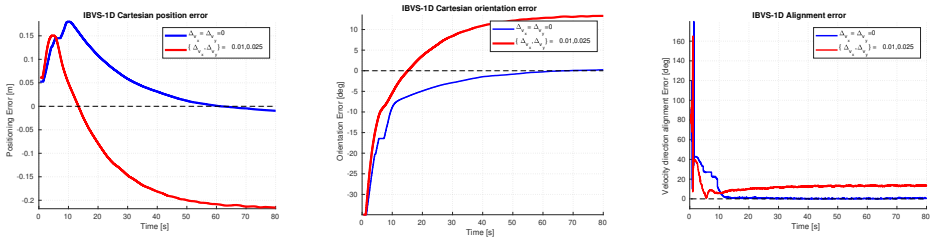
being $\boldsymbol{J}^{\#} = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1}$ the pseudo-inverse of $\boldsymbol{J}_x$ and $\boldsymbol{v}^* = [v\cos q_p, v\sin q_p, 0]^T$ a desired velocity, guarantees the exponential convergence of x_m to 0. In addition we observe that, spanning the null-space (2.27) with the coefficients $\{v,\ 0\}$, we also ensure that (2.12) is satisfied, at the regulation point, through the projection in the null-space $(\boldsymbol{I}_3 - \boldsymbol{J}_x^{\#}\boldsymbol{J}_x)$.

The first equation of (2.28) allows to directly compute the three Cartesian velocity inputs v_x, v_y and ω_z from the evaluation of the visual error $\boldsymbol{e}_v = x_m$. However, we remark that (2.27) holds only for $x_m = 0$, i.e., when the error dynamics has reached a stable equilibrium point. Under this condition, the control law (2.28) can be also written as

$$\begin{aligned} \mathrm{v}_r &= -\boldsymbol{J}_x^{\#} J_{\omega_z}^x u_p + \boldsymbol{v}^* \\ u_p &= -K_p e_p \end{aligned} \tag{2.29}$$

since the desired vector $\boldsymbol{v}^* \in \mathcal{N}(\boldsymbol{J}_x)$. When this is not true, i.e., during the transient response of the system, the basis of the null-space does not coincide with (2.27), and the projection in the null-space $(\boldsymbol{I}_3 - \boldsymbol{J}_x^{\#}\boldsymbol{J}_x)$ is required to not perturb the regulation of x_m. Figure 2.8 shows the control results of an omnidirectional wheeled robot, navigating within a straight corridor clear from obstacles. The figure shows that, by regulating the visual feature x_m to 0 on the image plane, the robot successfully achieves the navigation task, as both the Cartesian position error x and the orientation error θ_c converge to 0 (Figures 2.8a-2.8b). In addition, the alignment error ϕ between the motion direction and the camera principal axis is regulated to 0 as desired. However, it is worth to highlight that, since the control (2.28) is able to enforce (2.12) only when the visual task is regulated, the robot could move along hazardous trajectories that can increase the risk of collisions against undetected obstacles, (see Figure 2.15).

Figure 2.9 shows the same simulation results, compared with the perturbed scenario in which actuation disturbances act on the system. Specifically, assuming to simulate as actuation disturbance $\Delta v_x = 0.01$ m/s and $\Delta v_x = 0.025$ m/s, we observe that the control (2.28) is no more able to regulate the Cartesian pose of the robot along the bisector of the corridor (see Figure 2.9a-2.9b), that actually converges on a parallel of the bisector. Similarly, the alignment error ϕ converges to a non-zero value, resulting in a motion direction that persistently differs from the camera principal axis.



(a) **(b)** **(c)**

Figure 2.9. Simulation results for an omnidirectional wheeled robot navigating in a straight corridor under the 1D-IVBS + NSP control, considering as actuation disturbances $\Delta v_x = 0.01$ m/s and $\Delta v_y = 0.025$ m/s. (a) Position Error x. (b) Orientation error θ_c. (c) Alignment error ϕ.

2D-IBVS with Null-space projection

We consider, in this case, both the visual features x_m and y_m defined in (2.10). Therefore, we define the navigation task as $\boldsymbol{e}_v = (x_m, y_m)^T$, with dynamics given by (2.20). This choice is motivated by the observation that, when moving along the bisector of the corridor, the measured centroids are at the same height on the image plane. We can then consider the visual error vector $\boldsymbol{e}_v = [x_m, y_m]^T$.

The matrix $\boldsymbol{J}$ is now 2×3, so the dimension of the associated null-space $\mathcal{N}(\boldsymbol{J})$ is 1. In detail, one finds that, for $\boldsymbol{e}_v = 0$, a basis for the null-space is given by

$$\boldsymbol{B}_2 = \left\{ \begin{bmatrix} \cos q_p \\ \sin q_p \\ 0 \end{bmatrix} \right\} \tag{2.30}$$

that, as in the previous case, is the unit vector describing the direction of the principal axis of the camera in $\mathcal{F}_r$. So, in order to guarantee the exponential convergence of $\boldsymbol{e}_v$ to 0, we design the following control law

$$\mathrm{v}_r = -\boldsymbol{J}^{\#} \left(\boldsymbol{K} \boldsymbol{e}_v + \boldsymbol{J}_{\omega_z} u_p \right) + \left(\boldsymbol{I}_3 - \boldsymbol{J}^{\#} \boldsymbol{J} \right) \boldsymbol{v}^* \tag{2.31}$$

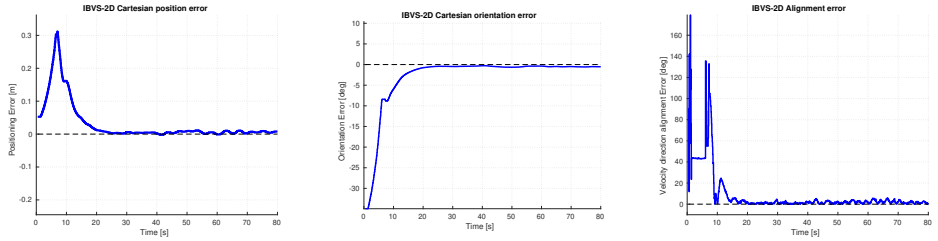
for $\boldsymbol{K} > 0$. As for (2.28), the controller successfully aligns the motion direction with the camera principal axis only if $\boldsymbol{e}_v = 0$. In this case, (2.31) simplifies as

$$\mathrm{v}_r = -\boldsymbol{J}^{\#} \boldsymbol{J}_{\omega_z} u_p + \boldsymbol{v}^* \tag{2.32}$$

Instead, for $\boldsymbol{e}_v \neq (0,0)^T$, the expression of $\boldsymbol{B}_2$ is not equivalent to (2.30) and the same considerations about the transient response, detailed for the controller 1D-IBVS + NSP, can be concluded.

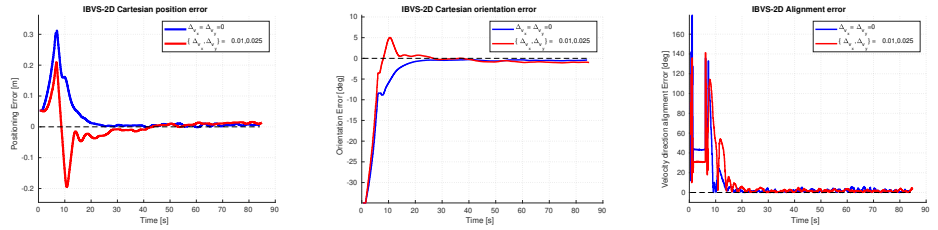
In Figure 2.10, simulation results of the wheeled robot in the straight corridor are shown. We observe that the robot pose is successfully regulated to the bisector of the corridor, and that the alignment error ϕ between the camera principal axis and the motion direction is zeroed through the null-space projection.

In addition, this control reveals to be robust to the presence of actuation disturbances, as shown in Figure 2.11a-2.11b: the Cartesian position and orientation of the robot is successfully regulated to zero despite the system actuates values of v_x



(a) **(b)** **(c)**

Figure 2.10. Simulation results for an omnidirectional wheeled robot navigating in a straight corridor under the 2D-IVBS + NSP control, in nominal conditions. (a) Position Error x. (b) Orientation error θ_c. (c) Alignment error ϕ.



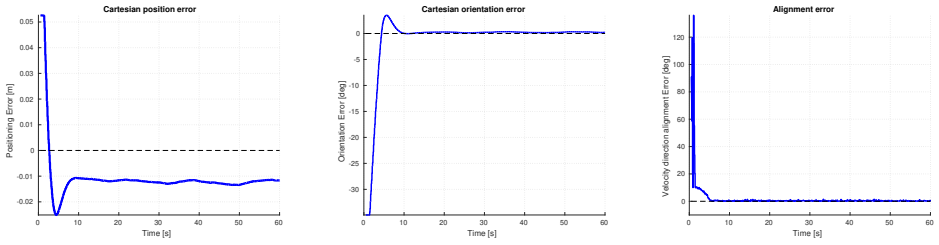
(a) **(b)** **(c)**

Figure 2.11. Simulation results for an omnidirectional wheeled robot navigating in a straight corridor under the 2D-IVBS + NSP control, considering as actuation disturbances $\Delta v_x = 0.01$ m/s and $\Delta v_x = 0.025$ m/s. (a) Position Error x. (b) Orientation error θ_c. (c) Alignment error ϕ.

and v_y that differs from the nominal ones. Therefore, the robot correctly brings to the center of the corridor, thus maximizing distances from the walls. This is also reflected in the plot of Figure 2.11c, where the alignment error ϕ is highlighted: the regulation of the motion direction towards the camera principal axis is not affected by the presence of actuation errors on the system, and the error is finally regulated to zero.

Nevertheless, during the transient response of the system, under the control (2.31), the robot may move along paths leading to collisions, as for the control 1D-IBVS + NSP, due to the null-space projector that violates (2.12) while still regulating $\boldsymbol{e}_v$. Indeed, the system exhibits abrupt and hasty trajectories, that could be motivated by the fictitious centroids positioning on the image borders, when no obstacles are detected on the image. Specifically, the workaround is critical and significantly alters the dynamics of y_m. This issue is enhanced by the fact that the image features extracted from the image (i.e., the obstacle centroids) do not really coincide with the same 3D space points, as they are computed based on the parts of obstacles appearing in the image. Therefore, while a regulation of the feature y_m is achieved at steady-state, it is not easy to study the behavior of the system along its transient response, as the dynamics used to describe the feature motion may not be consistent.

For this reason, we propose in the next paragraph a further control scheme that gets rid of the issues related to the transient response, by considering only the feature



(a) **(b)** **(c)**

Figure 2.12. Simulation results for an omnidirectional wheeled robot navigating in a straight corridor under the 1D-IVBS control with explicit assignment, in nominal conditions. (a) Position Error x. (b) Orientation error θ_c. (c) Alignment error ϕ.

x_m and actually exploiting the steady-state version of them, without recurring to a control scheme with null-space projector.

1D-IBVS with explicit assignment

Consider again the visual task given by $e_v = x_m$, with dynamics given by the first row of (2.20). To satisfy the constraint (2.12) at all times, we design a control scheme that consider an explicit assignment of (2.12) on the two Cartesian velocity inputs v_x and v_y. This actually excludes v_x and v_y to be assigned through the evaluation of the visual task, that is actually in charge of generating the proper angular velocity ω_z. Therefore, the control law is designed as follows

$$\begin{aligned} \omega_z &= (J^x_{\omega_z})^{-1}(-K_m x_m - J^x_{v_x} v_x - J^x_{v_y} v_y) - u_p \\ u_p &= -K_p e_p, \end{aligned} \tag{2.33}$$

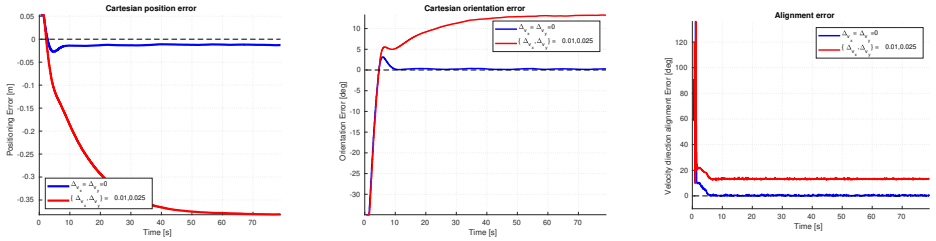
where v_x and v_y are assigned as (2.12), while K_m and K_p to denote positive control gains. Figure 2.12 shows the simulation results for this controller, for a wheeled robot moving along a straight and clear corridor. We observe that this controller successfully regulates the Cartesian pose on the corridor bisector (see Fig. 2.12). In addition, the control shows a short transient time to regulate the alignment error to 0 (Fig. 2.12c), due to the explicit assignment on v_x and v_y that satisfies (2.12). Indeed, in the nominal conditions in which the robot velocities are properly actuated, the settling time of ϕ depends only on the time required by the robot actuators to achieve the reference values (2.12).

However, when considering the presence of actuation disturbances in the system, this controller is not able to regulate the position and orientation errors to zero (see Fig. 2.13a-2.13b), since the constraint (2.12) is persistently violated, as no feedback information is taken into account to correct the error. As a result, the robot converges along a trajectory that is parallel to the corridor bisector.

In the following paragraph, we present a comparison analysis that highlights similarities and differences of the proposed controllers.

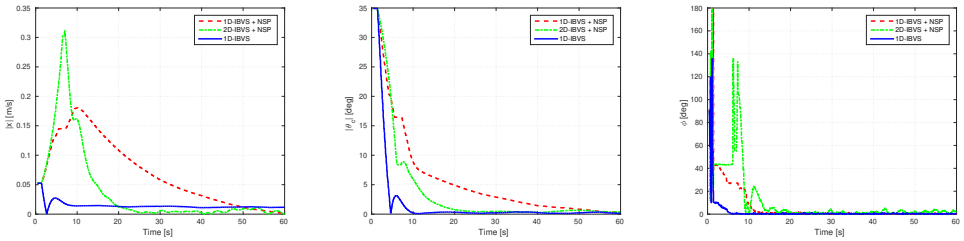
2.2.5 Discussion

In Figure 2.14, we reported the simulation results of the three proposed controllers, for an omnidirectional wheeled robot navigating in a straight corridor, in the ideal



(a) (b) (c)

Figure 2.13. Simulation results for an omnidirectional wheeled robot navigating in a straight corridor under the 1D-IVBS control, considering as actuation disturbances $\Delta v_x = 0.01$ m/s and $\Delta v_x = 0.025$ m/s. (a) Position Error x. (b) Orientation error θ_c. (c) Alignment error ϕ.

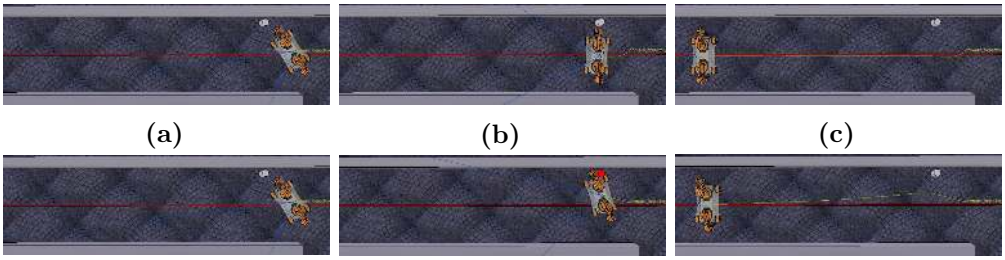


(a) (b) (c)

Figure 2.14. Comparison of regulation errors. From left to right: (a) Cartesian position error x; (b) Cartesian orientation error θ_c; (c) alignment error ϕ between motion direction and camera principal axis.

case in which the control inputs are correctly actuated by the system. While the regulation of errors is successfully achieved at steady-state as expected, it is worth to highlight the different behaviors of the controllers during the transient response of the system. Specifically, we observe that 1D-IBVS outperforms the controllers the 1D-IBVS+NSP and 2D-IBVS+NSP showing a lower settling time of the alignment error ϕ, thus reducing the possibilities to collide against unseen and unexpected obstacles. As already explained in the previous paragraph, this difference is due to the explicit assignment of the constraint (2.12) on the control inputs v_x and v_y, that forces the system to move in the desired direction independently from the regulation of the visual task. Formally, we can describe this behavior in terms of task-priority formulation. Indeed, the controller 1D-IBVS correspond to a controller that assigns the alignment constraint (2.12) a higher-order priority, while keeping the visual task as lower-order priority task, and solved in the null-space of (2.12). In this fashion, the controller 1-IBVS+NSP inverts the priorities of the two considered tasks, since the visual task is first solved through the pseudo-inverse of $\boldsymbol{J}$, leaving the alignment constraint as a lower-level priority task to be satisfied only when x_m has been regulated to 0.

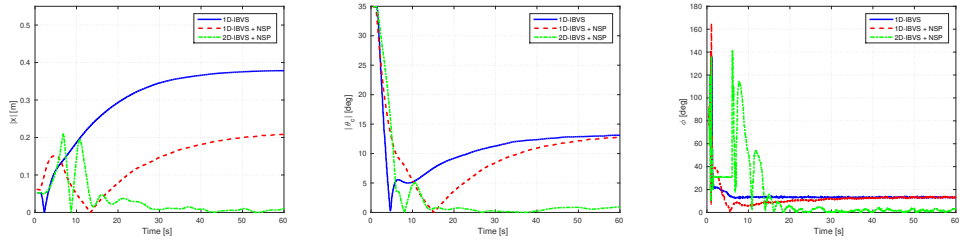
However, when we consider the presence of actuation disturbances in the system, we have already observed in the previous paragraph that 1-IBVS is no more able to regulate the Cartesian and the alignment errors to zero, with the effect that the



(a) (b) (c)

(d) (e) (f)

Figure 2.15. Transient response comparison, with an omnidirectional wheeled robot navigating in a straight corridor: (a)-(b)-(c) Safe trajectory generated under the control (2.33). (d)-(e)-(f) Unsafe trajectory generated under the control (2.28), due to the violation of the constraint (2.12) during the transient response.



(a) (b) (c)

Figure 2.16. Comparison of regulation errors when the system is affected by actuation disturbances. From left to right: (a) Cartesian position error x; (b) Cartesian orientation error θ_c; (c) alignment error ϕ between motion direction and camera principal axis. Here $\Delta v_x = 0.01$ m/s and $\Delta v_y = 0.025$ m/s.

robot converges towards a trajectory that is parallel to the corridor bisector. In this case, in fact, only the control 2D-IBVS+NSP successfully regulates the Cartesian position and orientation to zero, along with the alignment error ϕ.

This behavior is due to the task augmentation considered for this controller, where the feature y_m is taken into account as additional visual information. Indeed, this allows to balance the visual and the Cartesian tasks, so as to satisfy two control objectives in the image space (the features x_m and y_m) with two control objectives defined in the workspace (the position x and the orientation θ_c). However, the longer settling time of the alignment error ϕ still makes the system prone to possible collisions with unseen obstacles, and the safety in the navigation task could be compromised.

Given all these considerations, we propose a further scheme that takes the advantages of the proposed controllers. In detail, observing that the constraint (2.12) should have a higher priority with respect to the visual task, to guarantee safety in the navigation task, we consider a formulation of the visual controller as given by (2.33). However, to take into account also the presence of actuation errors, that both 1D-IBVS controls are not able to handle, we consider a direct velocity error by relying on an estimation of the robot velocity. Then, an external control loop is closed on the linear velocity components, in order to ensure that the alignment

constraint between the motion direction and the robot gaze is satisfied even when v_x and v_y are not properly actuated.

In the next Section, we describe the Kalman Filter observer, designed to estimate the robot velocity.

2.3 Kalman Filter for Velocity estimation

In this section, we describe the derivation of the Kalman Filter designed for the estimation of the navigation velocity of the robot. The filter fuses data coming from the proprioceptive sensors (IMU, encoders), with an optical flow-based velocity measurement, extracted from the image. In particular, the acceleration data from the IMU are used for the prediction step of the filter, in order to propagate the estimation of the robot velocity. Then, the update step refines the estimation through the velocity measurements provided by the remaining sensors: a vision-based velocity estimation method takes advantage of the optical flow computed for the reconstruction of the dominant plane, in order to generate a velocity measurement of the camera, while the encoders provide information about the measurement of the velocity generated through differential kinematics[1][26]. In detail, our goal is to estimate the vector $\boldsymbol{\xi} = [\mathbf{v}_I^T \ \boldsymbol{\omega}_I]^T$ of the linear and angular velocity, expressed in the reference frame attached to the IMU of the robot, $\mathcal{F}_I$. Considering the accelerometer data $\boldsymbol{u} = \boldsymbol{a}_I$ as the input of the system, the discrete-time stochastic propagation model of the state vector $\boldsymbol{\xi}$ can be written as

$$\boldsymbol{\xi}_{k+1} = \boldsymbol{\xi}_k + \Delta T \begin{bmatrix} \boldsymbol{I}_3 \\ \mathbf{0}_3 \end{bmatrix} \boldsymbol{u}_k + \boldsymbol{n}_k \tag{2.34}$$

with $\boldsymbol{n}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{N}_k)$ is a Gaussian random variable of the process noise with zero mean and covariance matrix $\boldsymbol{N}_k$. Since direct measurements of the robot velocity are available from vision and differential kinematics, the measurement model describing the predicted output of the system is simply

$$\boldsymbol{y}_k = \begin{bmatrix} \boldsymbol{I}_6 \\ \boldsymbol{I}_6 \end{bmatrix} \boldsymbol{\xi}_k + \bar{\boldsymbol{w}}_k \tag{2.35}$$

where $\bar{\boldsymbol{w}}_k = [\boldsymbol{w}_{VIS} \ \boldsymbol{w}_{DK}]^T \sim \mathcal{N}(\mathbf{0}, \bar{\boldsymbol{W}}_k)$ is the Gaussian random variable with zero mean and covariance matrix

$$\bar{\boldsymbol{W}}_k = \begin{bmatrix} \boldsymbol{W}_{VIS,k} & \mathbf{0}_6 \\ \mathbf{0}_6 & \boldsymbol{W}_{DK,k} \end{bmatrix} \tag{2.36}$$

being $\boldsymbol{w}_{VIS,k} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{W}_{VIS,k})$ and $\boldsymbol{w}_{DK,k} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{W}_{DK,k})$. From (2.34) and (2.35), the standard equations of the Kalman Filter can be applied for the estimation of the state vector $\hat{\boldsymbol{\xi}}_k$ at time k, as well as its associated covariance matrix $\boldsymbol{P}_k$ [27].

In the reminder of this section, we briefly describe the vision-based velocity estimation method adopted to generate the velocity measurement from the camera.

[1]The filter designed this way is meant to be general for any omnidirectional robot platform. However, the derivation of the measurement of the velocity through differential kinematics changes based on the type of robot adopted for the navigation task.

2.3.1 Velocity estimation based on optical flow

The visual measurement of the robot velocity is obtained through an optical-flow-based estimation method presented in [28] and also resumed in [29]. The algorithm considers the apparent motion of the camera pixels measured by the optical flow, to extract a subset of motion vectors that can consistently describe the motion of the camera. The subset is obtained through a pipeline of outlier rejection criteria, and is initialized with the dominant plane mask detected during the visual processing phase[2], in order to consider only pixels lying on the ground with known depth. The considered rejection criteria are:

- the pixel lies on the edges of the ground (detected by Canny operator);
- the motion vector direction points downwards;
- the motion vector magnitude stays within a lower and an upper bound;
- the u- and v-components of the motion vector stays within a confidence interval defined by the average and the standard deviation of all the motion vectors on the left and right sides of the image.

The subset of remaining motion vectors is used to generate a consistent measurement of the camera velocity (and so, of any other reference frame attached to the robot structure) by solving the following least-square problem:

$$\boldsymbol{v}_c = \arg\min_{\chi} \|\hat{\boldsymbol{L}}_{\chi}^{c}\boldsymbol{v} - \hat{\dot{\boldsymbol{x}}}_p\|_2 - \dot{\boldsymbol{q}}_p \tag{2.37}$$

where $\hat{\boldsymbol{L}}_{\chi}$ and $\hat{\dot{\boldsymbol{x}}}\boldsymbol{\prime}$ are built by stacking the interaction matrix $\boldsymbol{L}_i$ and the motion vector $\dot{\boldsymbol{x}}$ of each inlier pixel, respectively. Since we are interested in reconstructing the velocity of the robot body, the solution of (2.37) discards the contribution of the head pan joint velocity $\dot{\boldsymbol{q}}_p = [\mathbf{0}^T \ \dot{q}_p]^T$, that can produce erroneous measurements of the angular velocity ω_z when the head is rotating due to a non-zero u_p control input.

To conclude this section, we observe that, once the ego-motion of the robot is known, and since information about camera height and tilt is given, it is possible to predict the value of the optical flow of each pixel whose corresponding 3D point belongs to the ground. This is a redundant information that can be used to "double-check" the validity of the estimated ground plane. However, this aspect has not been discussed and it can be investigated in the future.

2.4 1D-IBVS with velocity estimation

In view of the issues related to the control schemes introduced in Sect. 2.2.5, and assuming to have a reliable estimation of the current robot velocity, we present a controller that directly evaluates the error between the desired linear velocity $\bar{\boldsymbol{v}}$ (given by (2.12), and the actual velocity $\boldsymbol{v}$ executed by the robot. This control

[2]Since the optical flow has been already computed for the dominant plane detection process, we can directly use it of this measurement process,thus saving computational time of the solution.

scheme is derived by considering an ideal model for the center of mass of the robot, i.e.,

$$\begin{cases} \dot{v}_x &= a_x \\ \dot{v}_y &= a_y \end{cases}. \tag{2.38}$$

Then, defining the error $\boldsymbol{e}_{xy} = [e_x, e_y]^T = [v_x - \bar{v}_x, v_y - \bar{v}_y]^T$, to guarantee exponential convergence to zero, we define the control inputs

$$\begin{aligned} a_x &= \dot{\bar{v}}_x - K_{v_x} e_x \\ a_y &= \dot{\bar{v}}_y - K_{v_y} e_y. \end{aligned} \tag{2.39}$$

with $k_{v_x} > 0$, $k_{v_y} > 0$. Finally, integrating both sides of (2.39) we get the corresponding velocity-level control law

$$\begin{aligned} v_x &= \bar{v}_x - K_{v_x} \int_0^t e_x(\tau) dt \\ v_y &= \bar{v}_y - K_{v_y} \int_0^t e_y(\tau) dt. \end{aligned} \tag{2.40}$$

i.e., the control assumes the nominal velocity $\bar{\boldsymbol{v}}$ as a feed-forward velocity, while zeroing the discrepancy between $\bar{\boldsymbol{v}}$ and $\boldsymbol{v}$ in terms of position drifting. In fact, the control law ensures to zero the error $\boldsymbol{e}_v$, thus guaranteeing the velocity direction and the robot gaze to be aligned. In the next Section, we show both simulation and experimental results to validate the effectiveness of the controller.

2.5 Simulations and Experiments

To validate the control scheme with velocity estimation, we performed simulations in the simulation environment V-REP, by employing both an omnidirectional wheeled platform and a humanoid robot NAO. A pair of experiments has also been performed on a real NAO robot.

In the following, we show the results obtained with the considered platforms.

2.5.1 Wheeled robot

To simulate an omnidirectional wheeled robot in the considered navigation conditions, we used an omnidirectional Youbot platform. Specifically, to show the possible mobility limitations due to a cumbersome structure, we added two manipulator arms on the top of the chassis, along with a monocular actuated camera mounted on the top of the platform and pointing downwards. This is a feasible and not limited setup, since, for particular applications, an omnidirectional robot with manipulation capabilities could be necessary and employed [30].

For the evaluated simulations, we placed the camera on the robot at 45 cm from the ground, oriented with a tilt angle $\gamma = 15°$. The acquired camera images have a resolution of 320×240. However, we considered a region of interest (ROI) of the source image to exclude detection of far objects (to avoid the robot reacts too fast) and close objects. This prevents the robot to react too fast and, at the same time, avoids that the projection of the arms on the image, when moving sideways, occludes the view of the environment and generates false detections. The

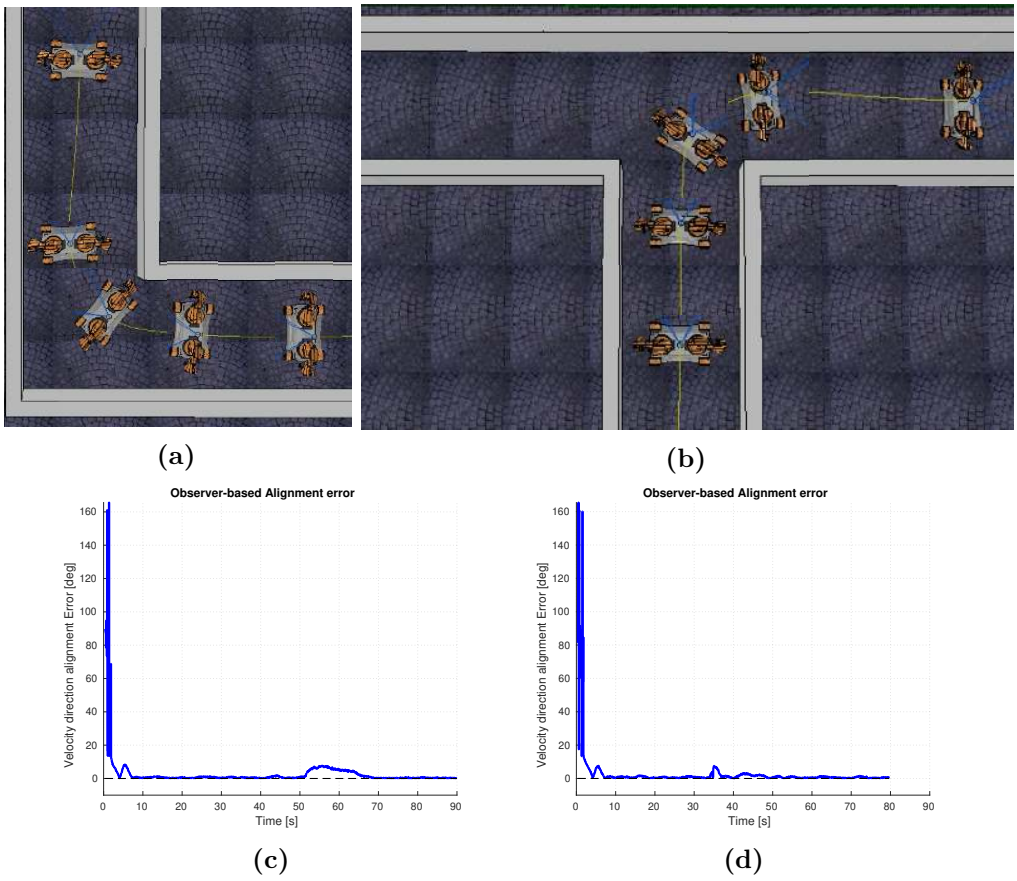


Figure 2.17. Two simulation scenarios with an omnidirectional wheeled robot: (a) negotiating a curve; (b) addressing a T-junction; (c)-(d) Alignment error ϕ in the two considered scenarios.

maximum linear velocity allowed for the robot is $v = 0.095$m/s. The thresholds for the evaluation of narrow passages are evaluated through an hysteresis mechanism and change between a pair of lower and upper values, to prevent false readings due to image processing noise. So we set $d_{min} = \{165, 180\}$ pxl and $d_{max} = \{185, 195\}$ pxl. For the estimation of the robot velocity, we only considered the vision-based velocity measurement for the update step of the filter. Then, the covariance matrices considered for the estimation are $\boldsymbol{N}_k = \text{diag}\left(10^{-10}, 10^{-10}, 10^{-10}, 10^{-1}, 10^{-1}, 10^{-1}\right)$, $\boldsymbol{W}_{VIS,k} = \text{diag}\left(10^{-9}, 10^{-9}, 10^{-8}, 10^{-8}, 10^{-8}, 10^{-8}\right)$. For the control law with velocity estimation, we set the control gains $K_{x_m} = 0.5$, $K_p = 0.45$, $K_{v_x} = K_{v_y} = 0.9$. Finally, we simulated an actuation disturbance on the linear velocity components by setting $\Delta v_x = 0.01$m/s and $\Delta v_x = 0.025$m/s.

We considered several navigation scenarios. In Figure 2.17, we show the trajectory performed by the robot during a curve negotiation and a T-junction, and the corresponding alignment errors ϕ evaluated during the simulations. Specifically, in Figure 2.17a, the robot moves along the bisector of the first part of the corridor, to maximise distances from the walls and guarateeing safety. When addressing the curve, the robot successfully detects the corner and turns in the direction of

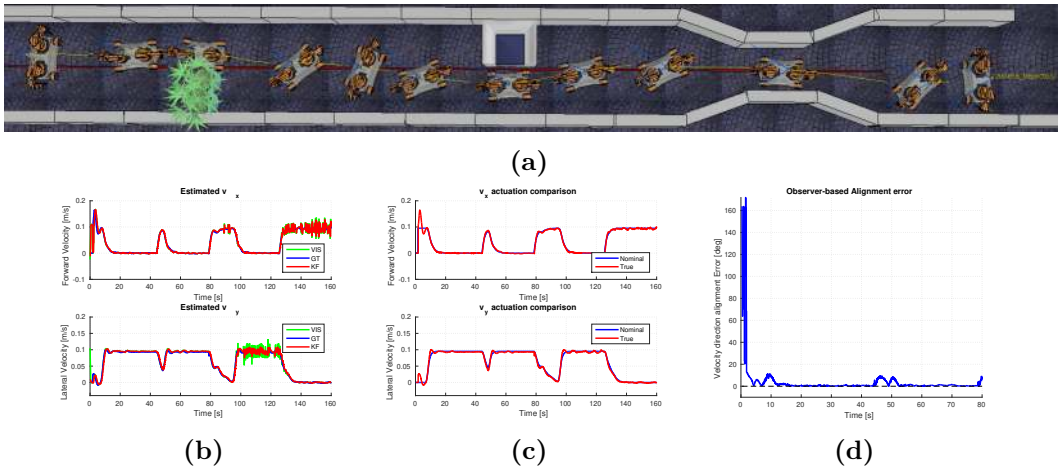
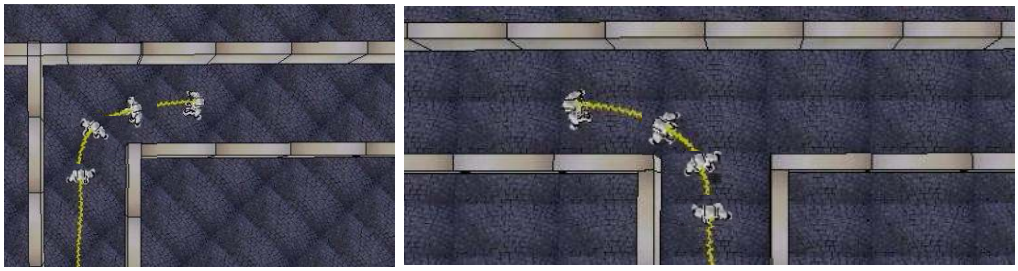


Figure 2.18. (a) Navigation scenario of a Youbot omnidirectional platform in simulated environment, where narrow passages are taken into account, due to the presence of obstacles along the path. (b) Comparison among the ground truth (GT) robot velocity (blue), the vision-based velocity measurement (green) and the velocity estimated with the Kalman Filter (red). (c) Comparison between the reference velocity (blue) given by (2.12) and the true robot velocity (red), after applying the control law (2.40). (d) Alignment error ϕ between the camera principal axis and the motion directions.

the second part of the corridor, avoiding collisions with walls during the turn and recovering the center of the corridor. On the other hand, Figure 2.17b shows the robot addressing a T-junction. Since there is not an explicit logic in the controller, when the robot reaches the junction it selects one of the two directions based on the appearance of the corridor in the image. At the T-junction a direction of motion could of course be commanded if a specific task would require the robot to move in a preferred direction.

Also in this case, the robot is able to recover the center of the corridor after finishing the curve. Figure 2.17c-2.17d show the alignment errors ϕ corresponding to the considered scenarios: we highlight that ϕ is properly regulated to zero even in presence of non-zero actuation disturbances. Moreover, the settling time to regulate the error to zero and align the motion direction with the camera principal axis is very short, thus the safety in the navigation is preserved.

In Fig. 2.18a, we considered a straight long corridor with a narrow passage and a pair of obstacles. The picture also shows the navigation pattern and the trajectory executed by the robot: whenever the robot detects a narrow passage upcoming along the path, the controller re-orients the robot by acting on the camera pan angle q_p, allowing the robot to safely cross the passage. In addition, the controller also takes into account actuation disturbances Δv_x and Δv_y injected in the system by estimating the current navigation velocity of the robot (Fig. 2.18b) and corrects properly the velocity commands through (2.40), in order to constantly satisfy (2.12). Fig. 2.18c shows the tracking of the true robot velocity with respect to the reference values, while Fig. 2.18d shows the alignment error ϕ between the camera principal axis and motion directions.



(a) (b)

Figure 2.19. Two navigation scenarios for a humanoid robot: (a) negotiating a turn, (b) addressing a T-junction.

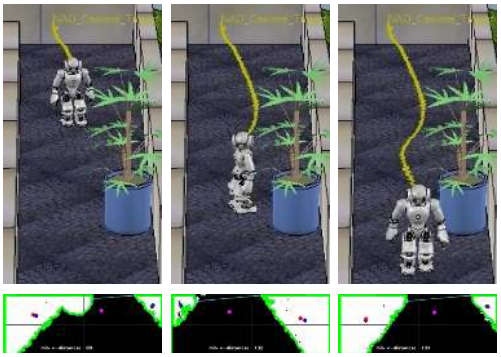


Figure 2.20. NAO navigates along a straight corridor and avoids an obstacle by walking sideways.

2.5.2 Humanoid robot

For the simulations with a humanoid robot, we used the small robot NAO. This robot is equipped with two cameras, mounted on the top and bottom of the forehead. Since the robot shoulder could occlude the field of view of the bottom camera when NAO turns the head, we chose to use the top camera. The optical flow has been computed through the Lucas-Kanade implementation available in the OpenCV open-source library. NAO is also equipped with a built-in walking engine in the NAOqi library, by which it is possible to send to the robot the linear and angular velocity command v_x, v_y and ω_z.

The main issue in implementing the proposed control scheme on a humanoid robot lies in the *sway motion* that typically affects the walking pattern of the robot. This has very drastic effects on all the vision-based measurement processes involved in the navigation scheme, as the acquired images can be blurred and the generated optical flow is prone to noise. While it is possible to identify and isolate the sway motion from the overall motion of the robot [31], satisfying results are also achieved by filtering out the generated oscillations in the affected signals [32]. Therefore, we applied a low-pass filter on the detected dominant plane image, as well as on the sensor measurements used for the estimation of the robot velocity. In addition, the oscillations also generate erroneous acceleration measurements, due to the projection of the gravity vector $\boldsymbol{g} = [0,\ 0,\ g]^T$ along the x- and y-axis of $\mathcal{F}_I$. We compensate such projections to generate reliable data for the Kalman Filter.

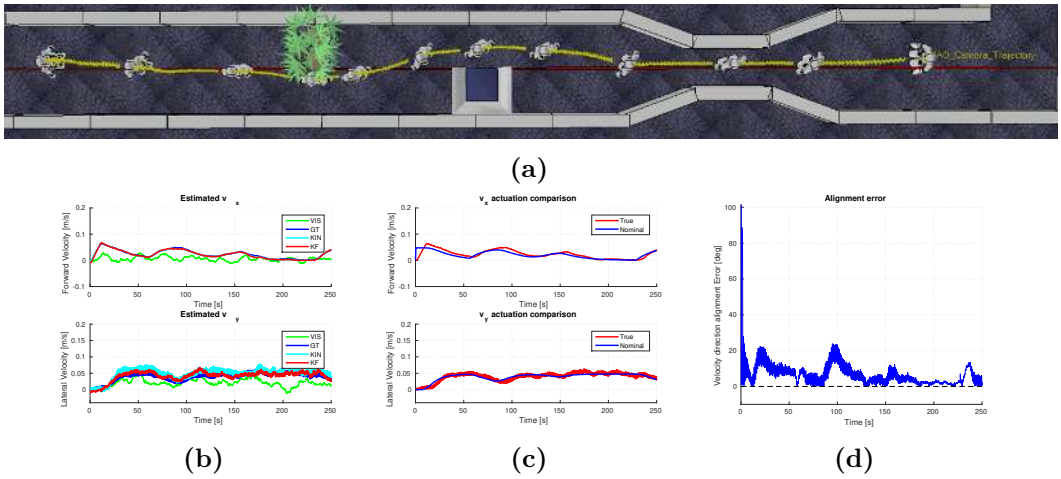
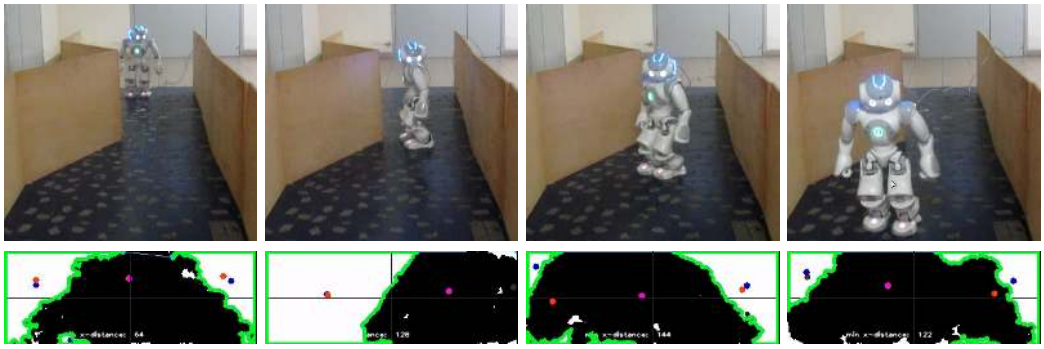


Figure 2.21. (a) Navigation scenario of a humanoid NAO robot in simulated environment, with the presence of obstacles along the path. (b) Comparison among the ground truth (GT) robot velocity (blue), the vision-based velocity measurement (green), the kinematics-based velocity measurement (cyan) and the velocity estimated with the Kalman Filter (red). (c) Comparison between the reference velocity (blue) given by (2.12) and the true robot velocity (red), after applying the control law (2.40). (d) Alignment error ϕ between the camera principal axis and the motion directions.

By measuring the torso angles α and β w.r.t. $\boldsymbol{g}$ through NAOqi, the compensated acceleration $\boldsymbol{a}$ is computed from the raw measurement $\hat{\boldsymbol{a}}$ as $\boldsymbol{a} = \hat{\boldsymbol{a}} - g\boldsymbol{b}$, where $\boldsymbol{b} = (\sin\beta \quad \cos\beta\sin\alpha \quad \cos\beta\cos\alpha)^T$.

In the considered simulation, we set $\gamma = 11.45°$. This angle allows a proper view on the scene, and also free motion of the pan angle q_p without risking collisions between the head and the shoulders. The maximum velocity norm allowed is $v = 0.0476$ m/s, while $d_{min} = \{100, 133\}$ pxl and $d_{max} = \{140, 145\}$ pxl. For the estimation of the robot velocity, we considered filtered vision- and kinematics-based velocity measurements in the update step of the Kalman Filter, with covariances matrices $\boldsymbol{W}_{VIS,k} = \text{diag}\left(10^{-6}, 10^{-6}, 10^{-6}, 10^{-7}, 10^{-7}, 10^{-7}\right)$ and $\boldsymbol{W}_{KIN,k} = \text{diag}\left(10^{-9}, 10^{-9}, 10^{-9}, 10^{-8}, 10^{-8}, 10^{-8}\right)$, respectively. The control gains are set to $K_{x_m} = 0.4$, $K_p = 0.05$, $K_{v_x} = K_{v_y} = 0.04$. Finally, we simulated an actuation disturbance by injecting a term $\Delta v_x = 10\% v$.

Also in this case, we considered several navigation scenarios. In Figure 2.19, we show the navigation scenarios of curve negotiation and T-junction addressing. On the other hand, in Figure 2.20 we tested the effectiveness of the navigation algorithm when obstacles other than a corridor walls are present in the environment. In particular, we put a plant in the middle of a corridor with non-parallel walls, to create a narrow passage that the robot has to cross by maximizing the clearance to the obstacles. The reference pan angle q_p^* is computed in accordance to eq. (2.11), where we set $t_l = \{90, 130\}$, $t_r = \{110, 140\}$. The robot starts walking off the corridor center, but it quickly converges to its bisector. Next, when the plant becomes visible in the image plane, the minimum distance d between left and right obstacles decreases and drops below t_l. The camera changes orientation with respect to the torso of the robot, that rotates to reduce the visual error, according the



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

Figure 2.22. Experiment with a real NAO. Top: snapshots of the robot moving in the corridor. Bottom: corresponding image procesisng. In the first snapshot a narrow passage is detected and used to set the desired value of q_p^* for camera rotation. When the camera orientation is equal to the desired value, the robot reaches the passage, as shown in the second snapshot. The third and the fourth snapshots show the robot re-aligning the camera with the sagittal plane, in correspondence of a wider free space.

control law (2.33). This induces the robot to walk sideways reducing the lateral encumbrance while traversing the narrow passage. When the obstacle disappears from the image, the minimum distance d exceeds t_h and the robot aligns the camera again with the torso, restoring the standard walking. This allows to successfully cross the narrow passage reducing the risks of collisions. When the robot is beyond the obstacle, the center of the corridor is recovered after the original configuration is accomplished.

A scenario with multiple obstacles, instead, is shown in in Fig. 2.21 with the corresponding simulation results. In Fig. 2.21a, we show the trajectory and the internal reconfiguration of the robot when approaching narrow passages: the control compensates the actuation disturbance estimated by the Kalman Filter (as shown in Fig. 2.21b) and allows to properly track the reference velocity determined trough (2.12)(shown in Fig 2.21c). The alignment error ϕ between the camera principal axis and the motion directions is shown in Fig. 2.21d.

2.5.3 Experiments with the humanoid robot NAO

Experiments to validate the filter and the control framework have been performed with the humanoid robot NAO. To give a quantitative measurement of the accuracy of the estimated velocity, and provide a highly reliable external measurement of the velocity as ground truth, we used a VICON Motion Capture System[3]. The ground truth data are generated by placing a number of markers on the surface of the robot head, denoting a new reference frame $\mathcal{F}_m$. Then, a least-square-based camera-VICON calibration procedure is run to estimate the homogeneous transformation matrix ${}^m\boldsymbol{T}_c$ between the the frame $\mathcal{F}_m$ (detected by the VICON system) and the robot camera frame $\mathcal{F}_c$.

The scenario reconstructed is similar to the one shown in the simulations, and adapted to the robot size. A straight corridor with a narrow passage has been

[3]https://www.vicon.com/

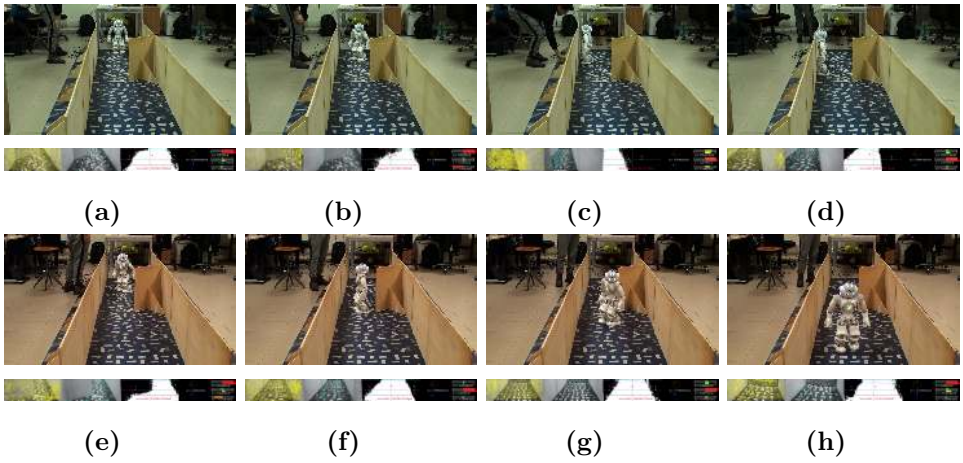
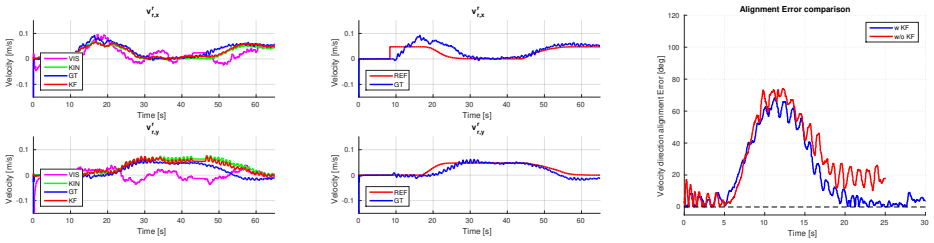


Figure 2.23. Experiments with the robot navigating in a straight corridor with a narrow passage, and a simulated perturbation $\Delta v_x = 20\% v_{\max}$. Top: snapshots of the navigation achieved without external control loop on the linear velocity components. Bottom: snapshots of the navigation achieved through velocity estimation and external control loop on the linear velocity components. On the bottom of each snapshots, the relevant steps of image processing have been highlighted: (i) optical flow field (yellow); (ii) inliers flow field for vision-based measurement (cyan); (iii) dominant plane image with x_l, x_r and x_m; (iv) command panel.

installed through wood panels and a patched carpet, as depicted in Figure 2.22-2.23. In Fig. 2.22, we showed the effectiveness of the control in nominal conditions, i.e., when no actuation disturbances are acting on the robotic system. A narrow passage is created at the beginning of the corridor, so the camera rotates to the desired value $q_p^* = 90^\circ$ when the robot begins to move. The direction of the gaze determines the direction of the driving velocity, so the robot moves sideways when the walls are closer. When the space between walls becomes wider the camera rotates to reach the desired value of $q_p^* = 0^\circ$ and realigns with the robot sagittal plane.

To simulate a disturbance on the actuation of the linear velocity, we explicitly added a term $\Delta v_x = 20\% v_{\max}$ to the command v_x sent to the robot. In Fig. 2.23, we highlight the effects of the external control loop based on the estimation of velocity: in (Fig. 2.23a-2.23b-2.23c-2.23d), the robot is navigating without taking into account the estimation of the current velocity. When the robot starts walking, the perturbation acts on the direction parallel to the camera principal axis, while it affects the perpendicular direction as the robot rotates to reconfigure itself. In this latter case, the constraint (2.12) is violated, and the robot is not moving along the camera principal axis direction.

On the other hand, when the robot moves under the 1D-IBVS control with velocity feedback (2.40) (see Fig. 2.23e-2.23f-2.23g-2.23h), that takes into account the current estimation of the robot velocity, the proper control inputs compensating the motion direction are computed, and the robot motion converges towards the reference values given by (2.12), thus reducing the alignment error of the motion direction with the camera principal axis to 0 (see Fig 2.24). As a result, the robot passes through the narrow passage without colliding with walls and recovers the center of the corridor.



(a) **(b)** **(c)**

Figure 2.24. Navigation results of a humanoid NAO robot in experimental session. (a) Comparison among the ground truth (GT) robot velocity (blue), the vision-based velocity measurement (magenta), the kinematics-based velocity measurement (green) and the velocity estimated with the Kalman Filter (red). (c) Comparison between the reference velocity (blue) given by (2.12) and the true robot velocity (red), after applying the control law (2.40). (d) Comparison of alignment errors ϕ between the first experiment without velocity estimation and correction (red) and the second one with velocity correction (blue).

2.6 Conclusions

In this chapter, we presented a vision-based navigation method for omnidirectional robots. Different visual schemes have been proposed, that maximize the the distances of the robot from obstacles by regulating suitable visual features in the camera image. We highlighted that, using the camera as only source of information, for the generation of the control inputs, is not enough to handle actuation disturbances or undesired transient response. Therefore, we further proposed a control scheme based on the estimation of the current robot velocity with a simple Kalman Filter, fusing data coming from the other on-board sensors of the robot. We showed the effectiveness of our method in simulation, both with a wheeled and a humanoid robot, and with an experiment, where a humanoid robot NAO is employed. Future works will consider dynamic environments where also moving obstacles are taken into account. In addition, the the heuristic logic adopted to change the reference q_p^* can also be robustified with a finite-state-machine. Finally, an accurate analysis of robustness on the system will be done, to quantitatively evaluate the effects of actuation disturbances on the system.

Chapter 3

Vision in Surgical Robotics

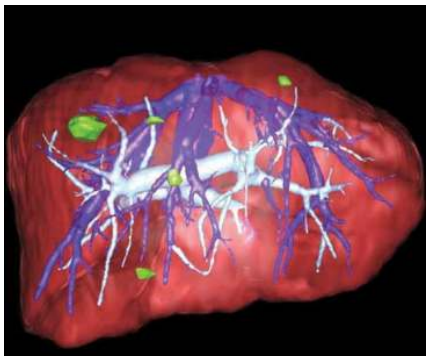


Figure 3.1. Reconstructed liver from patient CT scan image. Image retrieved from [33].

The wide expansion of robotic technologies in the recent years has encountered significant interest also in healthcare and medical contexts. The introduction of robotic systems and other related subjects in these fields (e.g., computer vision, control and state estimation problems, visual semantic reasoning, biomedical signal analysis, human-robot interaction) enabled a relevant growth in the applicability and in the performances of standard rehabilitative and surgical robotics.

Rehabilitative robotics concerns the employment of robotic systems and devices as active instruments demanded to the rehabilitation of patients, whose research has started in the late 80's [34][35][36][37][38]. During the latest years, several prototypes and devices have been continually developed, from robot-assisted powered wheelchairs [38], to arm-rehabilitative robotic systems [39], until to the modern and pioneering exoskeletons[40][41] and soft gloves [42].

While a strong contribution to this field has been given by research subjects concerning robot modeling, biomechanical signal analysis and force interaction, the deployment of visual data and related machine vision and vision-based algorithms has found wide application in clinical and surgical contexts, specifically for preoperative planning, intra-operative surgical procedures and postoperative evaluation as image-based therapies[43][44][45][46](Figure 3.1). Within this context, the major influence that robot technologies has exerted so far refers to the novel robot-assisted surgical procedures that brought to the definition of the Minimally Invasive Robotic Surgery (MIRS). MIRS refers to the set of clinical and surgical procedures that exploit the

Figure 3.2. da Vinci robotic system.

high accuracy and reliability of robotic systems to reduce the invasiveness of typical surgical procedures. This consists in reducing the size of incisions executed on the patient, thus minimizing the pain, the healing time of wounds and eventual clinical complications. For all these reasons, laparoscopic surgery is the field in which MIRS flourished most, thanks also to the development of the extensively employment of the da Vinci robotic system (Figure 3.2).

One of the great advantages in employing robotic systems for MIRS is related to the possibility to enhance the degree of autonomy for procedures that are still executed by surgeons, through the introduction of autonomous robot operation modes. This is not a straightforward problem because several procedures, performed by surgeons, are repetitive and tiring tasks, and can increase the operation time, costs and complications risks. Suturing is one of these procedures. From this point of view, automatic suturing presents many challenges related to both perception and manipulation. Left alone the interaction with soft tissues, an automatic suturing procedure has to deal with the difficulty of perception in an unstructured and highly noisy environment for needle and thread detection and tracking. The circular shape of the suturing needle encouraged the development of a family of *color-based* and *model-based* approaches for detection and tracking.

Therefore, needle detection and tracking are prerequisites to assist the suturing process, where information of interest can be retrieved from endoscopic images. Tracking the suturing needle is important because allows assistance functions for correct needle repositioning and surroundings knowledge. The task is challenging due to eventual occlusions that may occur during the procedure when treating tissue, moving camera, cluttered background and unconstrained motion.

A very common vision technique employed for image-based object detection is image segmentation [47]. In order to perform image-based needle detection, both available image source and needle type are crucial information to know. Percutaneous needle insertion, for instance, typically handle ultrasound (US) images or volumes, coming from a 2D or 3D US probe, for costs and safety reasons [48], [49], [50]. The needle type usually determines the adopted vision technique: Hough Transform (HT), for instance, is used for line detection in case of symmetrical-tipped needles, because they typically follow a straight path in the tissue [51], [52]. In [53], HT is used also for curved needle detection in 2D US images.

For automatic suturing, image source usually comes from an endoscopic camera system. However, most needle detection techniques are usually color-based [54], [55],

[56], [57], and detecting the needle tip instead of the full needle body could result in a failing approach. On the other hand, the drawback of color-based approaches is that they depend on the light conditions of the environment. Furthermore, reflections of the surroundings on the metallic surface of the needle may result in a varying needle color that is hard to identify with respect to the background.

For this reason, needle geometrical information can be taken into account and used as support for previous color-based detection inferences. Ellipse fitting is typically performed for this step, as 2D projection of 3D circles on the image plane [58], [59]. Geometry-based approaches are definitely more elegant and powerful, but they are computationally expensive from a pratical point of view.

The applications mentioned above can also be distinguished by considering the use of markers on the surgical instruments. While this can enhance the object detection in the image [56], [60], their use can be irrealistic, because surgical instruments are supposed to be autoclavable and alternatively used through a trocar. In any case, even markerless approaches assume at least to color the needle in advance, to ease the detection [57], [61]. Once the needle is detected in the image plane, tracking means to be able to refresh information about its position, based on - more synthetic - information coming from previous frames of the input video stream. For instance, region of interest (ROI) can be defined to reduce the overall computations for image segmentation [58], or needle pose estimation can be achieved through an external tracking device and projective geometrical considerations [59].

Authors in [62] present a RANSAC-based method for needle detection, where 3D needle pose reconstruction is achieved with the use of a stereo camera. However, the method does not run in real time and cannot be used for tracking. In [63], the 3D needle pose is adaptively reconstructed by relying on the observations of needle tip and junction, but tracking is not faced. Finally, [64] presents a method for a colored-needle tracking that involves a partial needle pose reconstruction and the use of markers. However, none of these works takes advantage of the kinematics information available from the robot, that are typically high-frequency and can ease the needle detection and tracking problem. Due to the circular shape of the suturing needle, a family of *model-based* approaches for detection and tracking are proposed in the literature. Some of them focus only on the needle recognition without taking into account the 3D pose estimation. E.g. in [62] and in [65], respectively, stereo and mono camera, in an unstructured environment, are used to track the needle in the image plane. On the other hand, in [63] and in [64], the 3D pose estimation is calculated using, respectively, an adaptive method based on the position of the two needle extremities (tip and junction point), and a method based on the needle geometry information in addition to the use of markers. This two approach have in common to consider a more structured environment respect to the previous. Furthermore, all these methods propose a purely vision-based needle recognition and tracking, without exploiting the high-rate information ($\geq$ 200 Hz) from robot kinematics on the pose of the end-effector holding the needle. This information can be used both to restrict the image region in which the detection can be performed and to provide a prediction of needle pose in the Cartesian space which benefits from the dynamical model knowledge of the needle grasped by the robotic surgical instrument.

In this chapter, we propose an approach for needle detection and tracking based

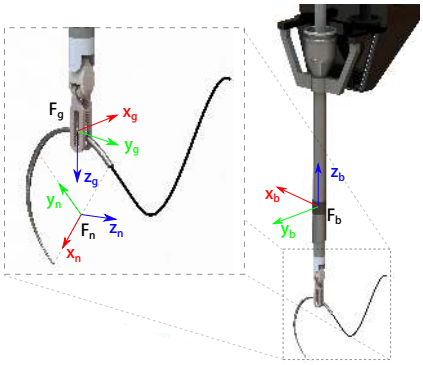


Figure 3.3. Reference frames.

on Kalman filtering, that combines visual information, extracted from a monocular camera, with the robot kinematics. Beside providing a fast and reliable needle pose estimation, the proposed method is robust with respect to scene variations as in case of partially needle occlusion or of needle re-grasping operation, as well as external disturbances perturbing the needle pose. In addition, the covariance matrices can be adapted taking into account the particular task that is being performed. In Section 3.1.2, we show some preliminary results for this problem.

3.1 Suturing Needle Tracking

In this section, we propose a state estimation scheme based on Extended Kalman Filter (EKF) [66], for the estimation of the pose of a suturing needle, held by a robot manipulator arm. Formally, with reference to Figure 3.3, we consider the needle pose $\mathcal{F}_n$, expressed in the base frame $\mathcal{F}_b$ of the manipulator holding the needle. The suturing needle has a typical semi-circular shape, designed to ease executing the required trajectories to perform stitching and suturing procedures.

The filter provides an estimate of the needle pose $\boldsymbol{\zeta} = [\boldsymbol{p}_n, \boldsymbol{q}_n]^T$, being $\boldsymbol{p}_n$ the true needle position, represented by the coordinates of the circle supporting the needle shape, and $\boldsymbol{q}_n$ its quaternion-based true orientation in $\mathcal{F}_b$. The prediction step provides a preliminary estimation of the needle pose through the linear and angular velocities of the gripper provided by the manipulator kinematics. Then, a vision-based 3D pose reconstruction is used in the filter correction step.

3.1.1 Extended Kalman Filter design

The linear and angular needle velocities $[\boldsymbol{v}_n, \boldsymbol{\omega}_n]^T$ are related to the linear and angular gripper velocities $[\boldsymbol{v}_g, \boldsymbol{\omega}_g]^T$, both expressed in $\mathcal{F}_b$, through the relationship

$$\begin{bmatrix} \boldsymbol{v}_n \\ \boldsymbol{\omega}_n \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_3 & -\lfloor \boldsymbol{r}_{gn\times} \rfloor \\ \boldsymbol{0}_3 & \boldsymbol{I}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_g \\ \boldsymbol{\omega}_g \end{bmatrix} \tag{3.1}$$

where $\boldsymbol{r}_{gn} = \boldsymbol{p}_n - \boldsymbol{p}_g$ is the relative position of the needle with respect to the gripper, expressed in $\mathcal{F}_b$, and $\lfloor \ * \ _{\times} \rfloor$ is the skew-symmetric matrix operator. From (3.1), we consider the following continuous-time process dynamics for the state vector $\boldsymbol{\zeta}$

$$\begin{array}{rl} \dot{\boldsymbol{p}}_n = \boldsymbol{v}_g + \left\lfloor \boldsymbol{\omega}_{g\times} \right\rfloor \boldsymbol{r}_{gn} + \boldsymbol{w}_p \\ \dot{\boldsymbol{q}}_n = \ \frac{1}{2}\boldsymbol{\Omega}({}^n\boldsymbol{\omega}_g)\boldsymbol{q}_n \ + \boldsymbol{w}_q \end{array} \tag{3.2}$$

where ${}^n\boldsymbol{\omega}_g$ is the angular gripper velocity expressed in $\mathcal{F}_n$, $\boldsymbol{w} = [\boldsymbol{w}_p, \boldsymbol{w}_q]^T \sim \mathcal{N}(0, \boldsymbol{W})$ is the process noise, and

$$\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix} -\lfloor \boldsymbol{\omega}_{\times} \rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^T & 0 \end{bmatrix}. \tag{3.3}$$

Pose measurements are retrieved through visual information extracted from a monocular camera. First, a detection algorithm computes the ellipse-shaped projection of the needle on the image plane. Then, its 3D pose is reconstructed given its size and the projection of a point (e.g., the tip [67]). So, The measurement model is given by

$$\boldsymbol{y} = \boldsymbol{\zeta} + \boldsymbol{m} \tag{3.4}$$

where $\boldsymbol{m} \sim \mathcal{N}(0, \boldsymbol{M})$ is the measurement noise. The corresponding error-state vector is defined as

$$\boldsymbol{\zeta} = \begin{bmatrix} \boldsymbol{p} & \delta\boldsymbol{\theta} \end{bmatrix}^T \tag{3.5}$$

where $\boldsymbol{p} = \boldsymbol{p} - \hat{\boldsymbol{p}}$ is the position error and $\delta\boldsymbol{\theta}$ is the 3×1 attitude error angle vector, defined by employing the small-angle approximation on the quaternion error

$$\begin{array}{rcl} \delta\boldsymbol{q} & = & \boldsymbol{q} \otimes \hat{\boldsymbol{q}}^{-1} \\ & = & \begin{bmatrix} \delta\boldsymbol{q}_{\mathrm{v}} \\ \delta q_s \end{bmatrix} \\ & = & \begin{bmatrix} \hat{\boldsymbol{k}} \sin(\delta\theta/2) \\ \cos(\delta\theta/2) \end{bmatrix} \\ & \approx & \begin{bmatrix} \frac{1}{2}\delta\boldsymbol{\theta} \\ 1 \end{bmatrix}. \end{array} \tag{3.6}$$

with $(\hat{\boldsymbol{k}}, \theta)$ being the corresponding axis-angle representation of the attitude error described by $\delta\hat{\boldsymbol{q}}$. The approximation is reasonable since the rotation can be assumed to be very small. The reason for the definition of the error vector (3.5) is explained by considering that only three of the four components of the quaternion vector are uncertain, since the fourth one is retrieved from the other three components through the unit norm constraint. As a consequence, from (3.2), (3.4), (3.5) and (3.6), we can derive the dynamics of the error vector and the measurement error as, respectively,

$$\begin{array}{rcrcl} \dot{\boldsymbol{p}} & = & \left\lfloor \boldsymbol{\omega}_{g\times} \right\rfloor \boldsymbol{p} & + & \boldsymbol{w}_p \\ \dot{\delta\boldsymbol{\theta}} & = & -\left\lfloor {}^n\boldsymbol{\omega}_{g\times} \right\rfloor \delta\boldsymbol{\theta} & + & \boldsymbol{w}_q \end{array} \quad = \quad \boldsymbol{f}(\boldsymbol{\zeta}) + \boldsymbol{w} \tag{3.7}$$

and

$$\boldsymbol{y} = \boldsymbol{\zeta} = \boldsymbol{h}(\boldsymbol{\zeta}) + \boldsymbol{m} \tag{3.8}$$

At this point, retrieving the jacobian matrices $\boldsymbol{F}_c$ and $\boldsymbol{H}_c$ from (3.7) and (3.8) is trivial, being

$$\begin{aligned} \boldsymbol{F}_c &= \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\zeta}}\right|_{\boldsymbol{\zeta}=\boldsymbol{\zeta}} = \begin{bmatrix} \lfloor \boldsymbol{\omega}_{g\times} \rfloor & \mathbf{0}_3 \\ \mathbf{0}_3 & -\lfloor {}^n\boldsymbol{\omega}_{g\times} \rfloor \end{bmatrix} \\ \boldsymbol{H}_c &= \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{\zeta}}\right|_{\boldsymbol{\zeta}=\boldsymbol{\zeta}} = \boldsymbol{I}_6 \end{aligned} \tag{3.9}$$

that are constant with respect to the current estimation $\boldsymbol{\zeta}$.

The equations (3.2),(3.4) and (3.9) are expressed with respect to continuous time variables. However, when the EKF has to be implemented, the corresponding discrete-time equations have to be taken into account. In the remainder of this document, we recap the the computations provided in [68]. So, considering a constant integration time step ΔT and an average input velocity $[\bar{\boldsymbol{v}}_g, \bar{\boldsymbol{\omega}}_g]$ between the time instant $t_k = k\Delta T$ and $t_{k+1} = (k+1)\Delta T$, we have the following discrete-time process dynamics

$$\begin{aligned} \boldsymbol{p}_{n,k+1} &= \boldsymbol{p}_{n,k} + \Delta T \left(\bar{\boldsymbol{v}}_g + \lfloor \bar{\boldsymbol{\omega}}_{g\times} \rfloor \boldsymbol{r}_{gn} \right) + \boldsymbol{w}_{p,k} \\ \boldsymbol{q}_{n,k+1} &= \boldsymbol{\Theta}({}^n\boldsymbol{\omega}_{g,k}, {}^n\boldsymbol{\omega}_{g,k+1}, {}^n\bar{\boldsymbol{\omega}}_g)\boldsymbol{q}_{n,k} + \boldsymbol{w}_{q,k} \end{aligned} \tag{3.10}$$

where

$$\boldsymbol{\Theta} = \left(\exp\left(\frac{1}{2}\boldsymbol{\Omega}({}^n\bar{\boldsymbol{\omega}}_g)\Delta T \right) + \frac{1}{48}\left(\boldsymbol{\Omega}({}^n\boldsymbol{\omega}_{g,k+1})\boldsymbol{\Omega}({}^n\boldsymbol{\omega}_{g,k}) - \boldsymbol{\Omega}({}^n\boldsymbol{\omega}_{g,k})\boldsymbol{\Omega}({}^n\boldsymbol{\omega}_{g,k+1}) \right)\Delta T^2 \right) \tag{3.11}$$

being

$$\exp\left(\frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega})\Delta T \right) = \begin{cases} \boldsymbol{I}_4 + \frac{\Delta T}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}), & \text{if} \quad |\boldsymbol{\omega}| \to 0 \\ \cos\left(\frac{|\boldsymbol{\omega}|}{2}\Delta T \right)\boldsymbol{I}_4 + \frac{1}{|\boldsymbol{\omega}|}\sin\left(\frac{|\boldsymbol{\omega}|}{2}\Delta T \right)\boldsymbol{\Omega}(\boldsymbol{\omega}), & \text{otherwise.} \end{cases} \tag{3.12}$$

The measurement model is simply written as

$$\boldsymbol{y}_k = \boldsymbol{\zeta}_k \tag{3.13}$$

Finally, the discrete expression of the jacobian matrices $\boldsymbol{F}_c$ and $\boldsymbol{H}_c$ are given by

$$\begin{aligned} \boldsymbol{F}_d &= \exp\left(\boldsymbol{F}_c\Delta T\right) = \begin{bmatrix} \exp\left(\lfloor \boldsymbol{\omega}_{g\times} \rfloor \Delta T \right) & \mathbf{0}_3 \\ \mathbf{0}_3 & \boldsymbol{\Phi} \end{bmatrix} \\ \boldsymbol{H}_d &= \boldsymbol{I}_6 \end{aligned} \tag{3.14}$$

where

$$\boldsymbol{\Phi} = \begin{cases} \boldsymbol{I}_3 - \Delta T\lfloor \boldsymbol{\omega}_{g\times} \rfloor + \frac{\Delta T^2}{2}\lfloor \boldsymbol{\omega}_{g\times} \rfloor^2, & \text{if} \quad |\boldsymbol{\omega}| \to 0 \\ \cos\left(|\boldsymbol{\omega}_g|\Delta T\right)\boldsymbol{I}_3 - \sin\left(|\boldsymbol{\omega}_g|\Delta T\right)\left\lfloor \frac{\boldsymbol{\omega}_g}{|\boldsymbol{\omega}_g|}_{\times} \right\rfloor + \left(1 - \cos\left(|\boldsymbol{\omega}_g|\Delta T\right)\right)\frac{\boldsymbol{\omega}_g}{|\boldsymbol{\omega}_g|}\frac{\boldsymbol{\omega}_g}{|\boldsymbol{\omega}_g|}^T, & \text{otherwise.} \end{cases} \tag{3.15}$$

At this point, we are ready to implement the equations of the EKF.

Prediction

The state prediction is obtained by propagating the state estimation $\hat{\boldsymbol{\zeta}}$ through (3.10)

$$\begin{array}{rcl} \hat{\boldsymbol{p}}_{n,k+1|k} & = & \hat{\boldsymbol{p}}_{n,k} + \Delta T \left(\bar{\boldsymbol{v}}_g + \left\lfloor \bar{\boldsymbol{\omega}}_{g\times} \right\rfloor \boldsymbol{r}_{gn} \right) \\ \hat{\boldsymbol{q}}_{n,k+1|k} & = & \boldsymbol{\Theta}({}^n\boldsymbol{\omega}_{g,k}, {}^n\boldsymbol{\omega}_{g,k+1}, {}^n\bar{\boldsymbol{\omega}}_g)\hat{\boldsymbol{q}}_{n,k} \end{array} \tag{3.16}$$

while the prediction of the corresponding covariance matrix $\boldsymbol{P}$ is given by

$$\boldsymbol{P}_{k+1|k} = \boldsymbol{F}_{d,k}\boldsymbol{P}_k\boldsymbol{F}_{d,k}^T + \boldsymbol{F}_{d,k}\boldsymbol{W}_k\boldsymbol{F}_{d,k}^T\Delta T \tag{3.17}$$

Update

In order to correct the state and the covariance matrix, we use the measurement $[\bar{\boldsymbol{p}}_{n,k+1}, \bar{\boldsymbol{q}}_{n,k+1}]$ at time $k+1$ to compute the innovation as

$$\boldsymbol{\nu}_{k+1} = \begin{bmatrix} \bar{\boldsymbol{p}}_{n,k+1} - \hat{\boldsymbol{p}}_{n,k+1|k} \\ \delta\bar{\boldsymbol{\theta}} \end{bmatrix} \tag{3.18}$$

where $\delta\bar{\boldsymbol{\theta}}$ is the 3×1 small-angle approximation of the quaternion measurement error

$$\begin{array}{rcl} \delta\bar{\boldsymbol{q}} & = & \bar{\boldsymbol{q}}_{n,k+1} \otimes \hat{\boldsymbol{q}}_{n,k+1|k}^{-1} \\ & \approx & \begin{bmatrix} \frac{1}{2}\delta\bar{\boldsymbol{\theta}} \\ 1 \end{bmatrix} \end{array} \tag{3.19}$$

Then, by taking into account (3.14), the Kalman gain is computed as

$$\boldsymbol{R}_{k+1} = \boldsymbol{P}_{k+1|k}\left(\boldsymbol{P}_{k+1|k} + \boldsymbol{M}_{k+1}\right)^{-1} \tag{3.20}$$

Using (3.18) and (3.20), we compute the correction term as

$$\Delta\boldsymbol{\zeta}_{k+1} = \begin{bmatrix} \Delta\boldsymbol{p}_{k+1} \\ \Delta\boldsymbol{\theta}_{k+1} \end{bmatrix} = \boldsymbol{R}_{k+1}\boldsymbol{\nu}_{k+1} \tag{3.21}$$

Finally, we can compute the corrected estimation $\hat{\boldsymbol{\zeta}}_{k+1}$ of the state at time $k+1$

$$\begin{array}{rcl} \hat{\boldsymbol{\zeta}}_{k+1} & = & \begin{bmatrix} \hat{\boldsymbol{p}}_{n,k+1} \\ \hat{\boldsymbol{q}}_{n,k+1} \end{bmatrix} \\ & = & \begin{bmatrix} \hat{\boldsymbol{p}}_{n,k+1|k} + \Delta\boldsymbol{p}_{k+1} \\ \boldsymbol{q}_{up,k+1} \otimes \hat{\boldsymbol{q}}_{n,k+1|k} \end{bmatrix} \end{array} \tag{3.22}$$

where, defining $\delta\boldsymbol{q}_{up,k+1} = \frac{1}{2}\Delta\boldsymbol{\theta}_{k+1}$, it is

$$\boldsymbol{q}_{up,k+1} = \begin{cases} \begin{bmatrix} \frac{\delta\boldsymbol{q}_{up,k+1}}{\sqrt{1-\delta\boldsymbol{q}_{up,k+1}^T\delta\boldsymbol{q}_{up,k+1}}} \end{bmatrix} & \text{, if } \delta\boldsymbol{q}_{up,k+1}^T\delta\boldsymbol{q}_{up,k+1} \leq 1 \\ \frac{1}{\sqrt{1+\delta\boldsymbol{q}_{up,k+1}^T\delta\boldsymbol{q}_{up,k+1}}} \begin{bmatrix} \delta\boldsymbol{q}_{up,k+1} \\ 1 \end{bmatrix} & \text{, otherwise} \end{cases} \tag{3.23}$$

and the corrected covariance matrix $\boldsymbol{P}_{k+1}$ is given by

$$\boldsymbol{P}_{k+1} = \boldsymbol{P}_{k+1|k} - \boldsymbol{R}_{k+1}\boldsymbol{P}_{k+1|k} \tag{3.24}$$

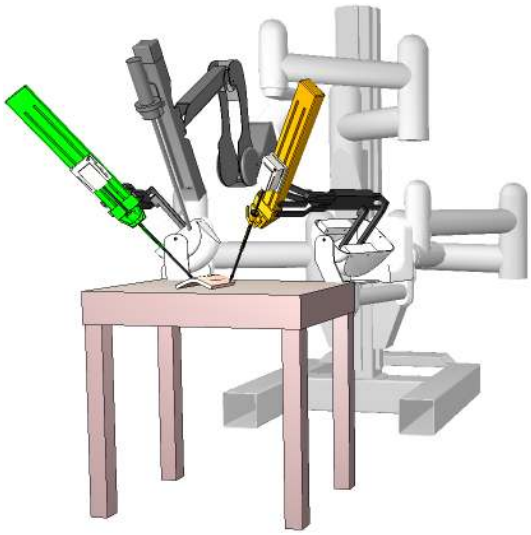


Figure 3.4. The da Vinci Research Kit simulator.

3.1.2 Simulation and Experimental Results

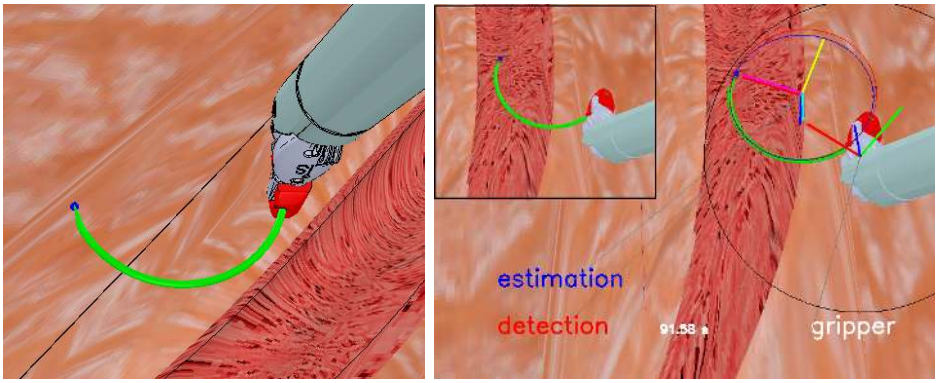
The filter described in the previous section has been validated both in simulations and experiments, considering a da Vinci robot surgical system. The obtained results are detailed in the following sections.

Simulation

To validate the presented filter in simulated environment, we exploited the da Vinci simulator developed for the V-REP virtual environment[69]. The simulator accurately reproduces the complete da Vinci system patient-side cart and provides the whole robot kinematics, namely Setup joint (SUJ), Patient Side Manipulators (PSMs), Endoscopic Camera Manipulator (ECM) (Figure 3.4). In this way, simulating the robot is a solution providing a low cost and easy to access environment for the development and proof of new control strategies while minimizing the risk of testing new algorithms on such a complex system. The entire simulator is designed to be fully integrated and interfaced with the open-source da Vinci Research Kit (dVRK) platform, and is programmed through the ROS framework. A detailed description of the simulator, along with extensions reproducing the Master surgeon console, can be found at the Appendix A of this manuscript.

To obtain the visual measurement of the needle pose required for the filter, a vision-based detection of the projected ellipse is achieved through a preliminary RGB-segmentation performed on a circular Region Of Interest (ROI). The ROI is centered at the gripper position, and its radius delimits the spherical region where the needle is supposed to be. Then, the set of pixels resulting from the segmentation is used to robustly fit the corresponding ellipse with a least-square-based approach. The detected ellipse is finally employed to recover a 6D pose measurement of the needle, based on geometric considerations [67].

The simulated setup is intentionally simple, and is depicted in the top left view in Fig. 3.5a: we considered a green-colored needle with a blue tip, to ehnance the

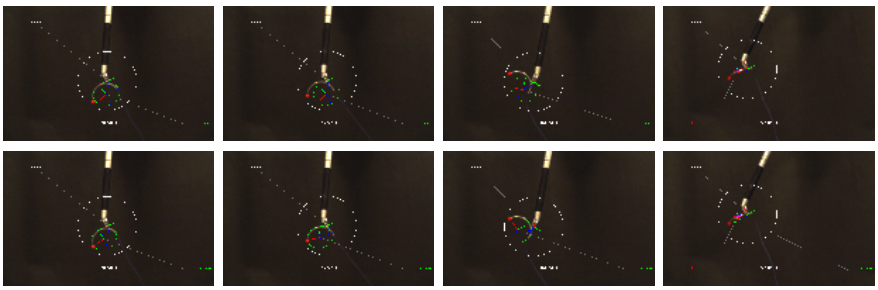


(a) (b)

Figure 3.5. (a) Simulation setup for the suturing needle tracking application. (b) Output of the image processing steps for the visual detection and the EKF-based estimation.

vision-based reconstruction of the proper needle orientation. Figure 3.5b shows the image processing output of the 3D needle tracking scheme: the circular ROI is drawn in black, while the output of the vision-based detection and the projection of the estimated pose of the needle are drawn in red and blue, respectively. Finally, the projections of the reference frames of the gripper, as well as of the vision-based reconstructed and the estimated pose, are drawn superposed on the image (RGB triad for estimation and gripper, CMY triad for the measurement).

Experiment



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

Figure 3.6. (a)-(b)-(c): Prediction failure scenario compared with the vision-based corrected estimation. (d): Detection failure scenario compared with the estimation. The white circle represents the image area in which the needle is assumed to be found, based on its radius and the depth of the gripper with respect to the camera.

To evaluate the robustness of needle pose estimation with respect to perturbations due to the needle-tissues interactions in real experiments, we used a real da Vinci robotic system, in a simplified experimental setup shown in Fig. 3.6. Typical vision-related challenges (e.g., shadows, sparkling metal surfaces, small-sized objects) are not considered to focus on the geometric part of the pose reconstruction. So, an RGB segmentation procedure and a least-square fitting are sufficient to extract the

projection of the needle on the image plane [70]. In addition, the needle tip has been colored to ease the detection of the projective point required for the 3D pose reconstruction.

The target experimental system is a da Vinci Research Kit robot (DVRK) [71]. Fig. 3.6 shows some preliminary results to prove the advantages in employing both robot kinematics and visual information for the needle pose estimation. During grasping, the needle is assumed as a rigid body attached to the gripper, and its pose can be predicted through robot kinematics, provided an initial estimate. However, since the needle-gripper transformation is not rigid, external disturbances (e.g., contact with tissues, slippages) can alter its pose, as shown in Fig (3.6a)-(3.6b)-(3.6c), where the needle pose has been explicitly changed. Robot kinematics can not cope with these disturbances, the prediction fails and propagates the error on the next iterations (top). The vision-based correction of our filter allows to detect the needle movements due to disturbances, and adjust the estimation accordingly (bottom). On the other hand, Fig (3.6d) shows a scenario where the vision-based detection fails, because the projected ellipse of the circular needle is degenerate on the image plane of the camera (top). However, kinematics information provided by the robot allows to maintain a stable estimation of the needle, even when this is not clearly visible. The figures are extracted from the videos that can be found at the link `http://www.diag.uniroma1.it/~labrob/research/ekfNeedleTracking.html`.

3.1.3 Conclusion

In this chapter we proposed an EKF-based approach to estimate the pose of the suturing needle during robot-assisted laparoscopic procedures. The filter fuses the kinematic information from the robot proprioceptive sensors with the visual information provided by a monocular endoscopic camera. The use of robot kinematics allows to restrict the region of interest in processing the image, thus speeding up computations, and renders the estimation robust with respect to visual occlusions. On the other hand, visual information allows to catch possible unmodeled motions of the needle, e.g., slippage or movements due to the interaction with tissues. Future work include development of robust image processing algorithms for the detection of the needle in realistic experimental setups. In vitro and ex-vivo experiments are planned to validate the approach in increasingly challenging conditions.

Chapter 4

Active Sensing for state estimation

In this manuscript, we presented two different applications where the state of a dynamic system is estimated through dynamic observers, using visual information. In the way in which they have been presented, the described Kalman Filters solved an observability problem, where the internal state of the system is reconstructed from the known inputs and outputs. In particular, the proposed observers are instances of a classical *passive* paradigm, where the system inputs are assumed to be given, and knowing the outputs is enough to guarantee the observability property for the system. However, for more complex dynamic non-linear systems, the reconstruction of the internal state may be not straightforward, since the observability property depends also on the inputs provided to the system. In particular, for a given non-linear system, there could exist some input trajectories that do not allow to fully reconstruct a reliable estimation of the state, making the system unobservable. In literature, these trajectories are referred as *singular inputs*.

To prevent the system from applying such inputs, the classical observability paradigm has to be reconsidered, to take into account the possibility to *actively* affect the inputs of the system. Modeling an estimation problem by enforcing optimality criteria on the system input trajectories is referred as *active sensing* or *active estimation*. Therefore, solving an active sensing problem is equivalent to analyze the observability property of the considered system. While the canonical Observability Rank (OR) provides a binary information about the observability of the system, there exist several mathematical tools that actually measure the degree of observability of the system with specific metrics. In [72], the active sensing control problem is addressed for nonlinear differentially flat systems. The proposed method considers a multiple-constraint optimization problem, to minimize on-line the maximum uncertainty of an EKF over a trajectory, by using the Observability Gramian. In a more recent work[73], the authors generalize the problem to take into account process and measurement noise, by exploiting the solution of the Continuous Riccati Equation. Authors in [74] address the problem of minimizing the maximum uncertainty of the pose estimation for a micro aerial vehicle, by designing an optimal path planner based on the photometric information acquired through the images of an equipped camera.

The active paradigm for state estimation is widely employed for Structure-from-Motion (SfM) problems. SfM is a vision-based technique aimed at estimating the 3D geometric structure of scene observed by a moving camera, starting from the 2D acquired visual information and the motion of the camera. While SfM is a well-established and known problem in computer vision, only recently novel frameworks addressing the problem in active sense have been proposed. A framework addressing the SfM problem in active fashion is proposed in [75], where the Observability Rank Condition (ORC) is exploited as metrics to evaluate the observability of the considered system. The presented framework is meant as a general method and can be applied to actively estimate different geometric primitives, e.g., 3D straight lines [76], points [77], spheres, cylinders[78] and planes[79][80]. In [81], the active SfM problem is solved by only using the linear velocity components. In other related works, active SfM schemes are managed to formulate a coupled framework with visual servoing control schemes [82][83][84].

In this chapter, we present a preliminary study of an active SfM problem for plane estimation, using the framework described in [75]. However, with respect to the related works [79][80], we do not use sparse visual data to reconstruct the planar scene, but we exploit a dense representation of the visual information.

As shown in Chapter 2, using dense visual data from the camera images allows to gather a huge amount of information, that can be used to robustly estimate geometric entities or the motion of the camera. Dense data are also exploited to design novel Visual Servoing (VS) schemes. Authors in [85][86] present an approach employing photometric information of all image pixels, taking into account the intensity values and the lighting model of the scene. While showing promising results in terms of accuracy, this method becomes intractable for large-size images. In addition, it relies on the computation of the spatial image gradient, that is typically an error-prone operator. Other possible direct dense VS schemes consider also wavelet[87], Gaussian mixtures[88] and depth maps[89].

Among the consistent related literature, an efficient representation of visual dense data that keep limited the size of the problem is given by weighted photometric moments [90]. With respect to the canonical and well-established image moments[24], weighted photometric moments are computed on the entire image and are robust to the appearance/disappearance of new portions of the scene in the image, thus enlarging the possible basin of convergence of the system.

However, for a visual servoing or a state estimation problem, choosing the most suitable set of moments is not a trivial issue. In [91], this problem is addressed by considering a weighted sum of image moments, whose weights are adaptively modified to select the best set of moments over time.

Within this perspective, the active SfM scheme proposed in this chapter considers weighted photometric moments to describe the visual dense information acquired by a monocular camera. In particular, we consider a weighted sum of moments to adaptively select the proper set of moments that maximize specific criteria on the estimation process. As a result, the considered measurement of the visual dense information is expressed as a weighted sum of weighted photometric moments.

4.1 Problem formulation

The addressed Structure-from-Motion problem considers the reconstruction of a planar scene, through the known motion of a moving camera. In this first formulation of the problem, we consider the camera as a free-flying object that can instantaneously move in any direction with velocity $\mathrm{v} = (\boldsymbol{v}^T, \boldsymbol{\omega}^T)^T$, without workspace limitations or non-holonomic constraints. The planar scene is formalized through the equation $\boldsymbol{n}^T\boldsymbol{X} + d = 0$, where $\boldsymbol{n}$ is the unit vector perpendicular to the plane, d is the distance of the plane from the camera and $\boldsymbol{X}$ represents a generic point lying on the plane.

The proposed SfM scheme, presented below, is an instance of the general framework shown in [75], where a function of weighted photometric moments is assumed as measurable component of the considered system for the estimation task.

The reminder of the chapter is organized as follows: Section 4.2 briefly summarizes the derivation of the dynamics of photometric moments given in [90]. Section 4.3 describes the general active SfM scheme used to estimate the geometric information of the scene. Section 4.4 shows some preliminary results.

4.2 Weighted Photometric moments

Visual servoing (VS) schemes based on photometric moments have been introduced to cope with the critical issues arising in standard geometric approaches. Indeed, these could be typically compromised when preliminary image processing steps (e.g., primitive extraction, tracking and matching), required to generate the visual features, are not reliable or effective. Photometric moments are based on the luminance information acquired by the camera image and, as for the geometric moments, reveal to be more efficient than methods that directly employ luminance data, due to the capability to synthetize dense information without significantly scaling-up the size of the problem.

As seen in Chapter 2, a significant problem in VS schemes is the limited field-of-view (FOV) of the camera, that reduces the convergence domain of the considered system and produces inconsistencies in the acquired visual data, due to the image portions that enter to or exit from the image plane. To cope with this phenomenon, the authors in [90] considers a weighting function in the classical formulation of photometric moment. Formally, the weighted photometric moment of order $p+q$ is defined as

$$m_{pq} = \int\int_{\pi} x^p y^q w(\boldsymbol{x})\, \boldsymbol{I}(\boldsymbol{x})\, dxdy \tag{4.1}$$

where π is the image plane and $\boldsymbol{x} = (x, y) \in \pi$ is a given pixel of the image, $\boldsymbol{I}(\boldsymbol{x})$ is the intensity level of the image in $\boldsymbol{x}$ and $w(\boldsymbol{x})$ is a weight associated to $\boldsymbol{x}$. As a visual feature, the derivative of m_{pq} can be related to the camera velocity $\boldsymbol{v}_c$, through the interaction matrix $\boldsymbol{L}_{m_{pq}}$:

$$\dot{m}_{pq} = \boldsymbol{L}_{m_{pq}} \mathrm{v}_c \tag{4.2}$$

with $\boldsymbol{L}_{m_{pq}} = \begin{bmatrix} L_{m_{pq}^{v_x}} & L_{m_{pq}^{v_y}} & L_{m_{pq}^{v_z}} & L_{m_{pq}^{\omega_x}} & L_{m_{pq}^{\omega_y}} & L_{m_{pq}^{\omega_z}} \end{bmatrix}^T$. Assuming image brightness

constancy and that the considered scene is planar, the entries of $\boldsymbol{L}_{m_{pq}}$ are given by

$$\begin{array}{rcc}
L_{m_{pq}^{v_x}} & = & Am_{p+1,q}^{\nabla x} + Bm_{p,q+1}^{\nabla x} + Cm_{p,q}^{\nabla x} \\
L_{m_{pq}^{v_y}} & = & Am_{p+1,q}^{\nabla y} + Bm_{p,q+1}^{\nabla y} + Cm_{p,q}^{\nabla y} \\
L_{m_{pq}^{v_z}} & = & -Am_{p+2,q}^{\nabla x} - Bm_{p+1,q+1}^{\nabla x} - Cm_{p+1,q}^{\nabla x} - Am_{p+1,q+1}^{\nabla y} - Bm_{p,q+2}^{\nabla y} - Cm_{p,q+1}^{\nabla y} \\
L_{m_{pq}^{\omega_x}} & = & -m_{p+1,q+1}^{\nabla x} - m_{p,q}^{\nabla y} - m_{p,q+2}^{\nabla y} \\
L_{m_{pq}^{\omega_y}} & = & m_{p,q}^{\nabla x} + m_{p+2,q}^{\nabla x} + m_{p+1,q+1}^{\nabla y} \\
L_{m_{pq}^{\omega_z}} & = & -m_{p,q+1}^{\nabla x} + m_{p+1,q}^{\nabla y}
\end{array} \tag{4.3}$$

where $(A, B, C)^T = -\boldsymbol{n}/d$ parametrize the pose of the observed plane with respect to the camera, and

$$\begin{array}{rcl}
m_{p,q}^{\nabla x} & = & \int\int_{\boldsymbol{\pi}} x^p y^q w\left(\boldsymbol{x}\right) I_x dxdy \\
m_{p,q}^{\nabla y} & = & \int\int_{\boldsymbol{\pi}} x^p y^q w\left(\boldsymbol{x}\right) I_y dxdy
\end{array} \tag{4.4}$$

being I_x and I_y the two components of the spatial gradient of the image. Since the image gradient computation is an error-prone step, applying the Green's Theorem, the expressions (4.4) is written without the dependence from I_x and I_y as

$$\begin{array}{rcl}
m_{p,q}^{\nabla x} & = & -pm_{p-1,q} - \int\int_{\boldsymbol{\pi}} x^p y^q \dfrac{\partial w}{\partial x} I\left(\boldsymbol{x}\right) dxdy + \oint_{\partial\pi} x^p y^q w\left(\boldsymbol{x}\right) I\left(\boldsymbol{x}\right) dy \\
m_{p,q}^{\nabla y} & = & -qm_{p,-1} - \int\int_{\boldsymbol{\pi}} x^p y^q \dfrac{\partial w}{\partial y} I\left(\boldsymbol{x}\right) dxdy - \oint_{\partial\pi} x^p y^q w\left(\boldsymbol{x}\right) I\left(\boldsymbol{x}\right) dx.
\end{array} \tag{4.5}$$

This way, the expression of the interaction matrix depends only on the form of the weighting function $w(\boldsymbol{x})$. Specifically, for $w\left(\boldsymbol{x}\right) = 1\ \forall \boldsymbol{x}$, assuming that it is $I\left(\boldsymbol{x}\right) = 0$ on the boundary of the image, substituting (4.5) in (4.3) results in the following entries of the interaction matrix:

$$\begin{array}{rcc}
L_{m_{pq}^{v_x}} & & -A\left(p+1\right)m_{p,q} - Bpm_{p-1,q+1} - Cpm_{p-1,q} \\
L_{m_{pq}^{v_y}} & & -Aqm_{p+1,q-1} - B\left(q+1\right)m_{p,q} - Cqm_{p,q-1} \\
L_{m_{pq}^{v_z}} & = & A\left(p+1+3\right)m_{p+1,q} + B\left(p+q+3\right)m_{p,q+1} + C\left(p+q+2\right)m_{p,q} \\
L_{m_{pq}^{\omega_x}} & & qm_{p,q-1} + \left(p+q+3\right)m_{p,q+1} \\
L_{m_{pq}^{\omega_y}} & & -pm_{p-1,q} - \left(p+q+3\right)m_{p+1,q} \\
L_{m_{pq}^{\omega_z}} & & pm_{p-1,q+1} - qm_{p+1,q-1}
\end{array} \tag{4.6}$$

However, this formulation requires that the image has a uniformly colored black background. This can be a strong limitation, also because the appearance/disappearance of new image portions cannot be handled.

To take into account this requirement, a non uniform weighting function is selected. Specifically, for $K > 0$ and $a > 0$, the weighting function

$$w\left(x, y\right) = K \exp^{-a\left(x^2+y^2\right)^2} \tag{4.7}$$

assigns higher weights to a centered circular area of image pixels, while exponentially decreasing values are assigned approaching to the image borders, in which they are assumed equal to 0 (see Figure 4.1). The resulting interaction matrix is given by

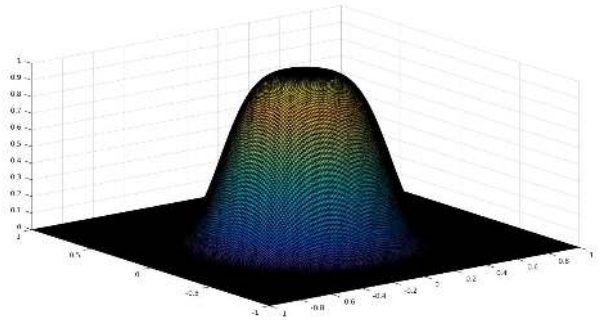


Figure 4.1. Graphical representation of the weighting function $w(x,y) = K\exp^{-a\left(x^2+y^2\right)^2}$.

$$\boldsymbol{L}_{m_{p,q}} = \boldsymbol{L}_{u_{p,q}} + 4a\boldsymbol{L}_{w_{p,q}} \tag{4.8}$$

where $\boldsymbol{L}_{u_{p,q}}$ is the matrix given in (4.6), and $\boldsymbol{L}_{w_{p,q}} = \begin{bmatrix} L^{v_x}_{w_{p,q}} & L^{v_y}_{w_{p,q}} & L^{v_z}_{w_{p,q}} & L^{\omega_x}_{w_{p,q}} & L^{\omega_y}_{w_{p,q}} & L^{\omega_z}_{w_{p,q}} \end{bmatrix}$ results in

$$\begin{array}{rcrl}
L^{v_x}_{w_{p,q}} & = & & 4aA\left(m_{p+4,q} + m_{p+2,q+2}\right) + \\
 & & + & 4aB\left(m_{p+3,q+1} + m_{p+1,q+3}\right) + \\
 & & + & 4aC\left(m_{p+3,q} + m_{p+1,q+2}\right) \\
L^{v_y}_{w_{p,q}} & = & & 4aA\left(m_{p+3,q+1} + m_{p+1,q+3}\right) + \\
 & & + & 4aB\left(m_{p+1,q+4} + m_{p+2,q+2}\right) + \\
 & & + & 4aC\left(m_{p,q+3} + m_{p+2,q+1}\right) \\
L^{v_z}_{m_{pq}} & = & - & 4aA\left(m_{p+5,q} + 2m_{p+3,q+2} + m_{p+1,q+4}\right) + \\
 & & - & 4aB\left(m_{p+4,q+1} + 2m_{p+2,q+3} + m_{p,q+5}\right) + \\
 & & - & 4aC\left(m_{p+4,q} + 2m_{p+2,q+2} + m_{p,q+4}\right) \\
L^{\omega_x}_{m_{pq}} & = & - & 4a\left(m_{p+4,q+1} + 2m_{p+2,q+3} + m_{p,q+3} + m_{p+2,q+1} + m_{p,q+5}\right) \\
L^{\omega_y}_{m_{pq}} & = & & 4a\left(m_{p+3,q} + m_{p+1,q+2} + m_{p+5,q} + 2m_{p+3,q+2} + m_{p+1,q+4}\right) \\
L^{\omega_z}_{m_{pq}} & = & & 0
\end{array} \tag{4.9}$$

From the expression above, we highlight that the interaction matrix of a weighted photometric moment $m_{p,q}$ of order (p,q) has a recursive definition that requires the knowledge of higher order moments up to the order $(p+q+5)$.

A general control or estimation scheme that uses this representation of visual information typically selects a proper set of photometric moments to accomplish the considered task. Nevertheless, for given input and output system trajectories, some orders of moments may be more significant than others. Therefore, the choice of the set of moments to use for the task becomes crucial.

Within this perspective, the choice of a linear combination of weighted photometric moments, with coefficients adaptively selected, is a reasonable solution to optimize the performances of the considered SfM problem [91], as shown in next section.

4.3 Active SfM scheme

In this section, we recapitulate and adapt the general active SfM scheme proposed in [75], to handle a set of weighted photometric moments and a corresponding linear combination of them. The goal of the active framework is to optimize the camera velocity and the linear coefficients, in order to find the most descriptive and informative set of weighted photometric moments, while generating the desired camera trajectory that optimizes specific criteria of the estimation process.

Specifically, we denote by $\boldsymbol{m}_w(\boldsymbol{\theta})$ a linear combination of moments up to a given fixed order (p, q), whose coefficients are collected in a vector $\boldsymbol{\theta}$:

$$\boldsymbol{m}_w(\boldsymbol{\theta}) = \boldsymbol{\theta}^T \boldsymbol{m}_{p,q} \tag{4.10}$$

where $\boldsymbol{m}_{p,q} = [m_{01}, m_{10}, m_{11}, m_{20}, \dots, m_{p,q}]^T$. Observe that the vector $\boldsymbol{m}_{p,q}$ of considered moments does not contain the zero-order moment m_{00}. This is a deliberate design choice since, in practice, m_{00} is around two orders of magnitude higher than other moments. Thus, considering m_{00} in $\boldsymbol{m}_{p,q}$ can unintentionally neglect the effects of all other moments, as its dynamics would be far too dominant.

This type of representation falls in the design choice presented in [91], considering a weighting function with a polynomial basis of fixed degree, where a maximum order of moments is chosen *a priori*. However, it is worth to highlight that the authors also propose a constrained polynomial basis, that takes into account additional constraints to guarantee that the resulting function zeroes at the image borders, along with its derivative. These boundary requirements allow to handle possible appearance/disappearance of image portions in/from the image plane, and needs a higher order polynomial basis to take into account the constraints on the borders. Nevertheless, we showed that this issue is successfully addressed by employing the weighted photometric moments (4.1). Therefore, we discard this latter design choice and keep a simpler solution with a fixed order polynomial basis.

The coefficients $\boldsymbol{\theta}$ play the role of weights, as they can be regulated to optimize some desired criteria in the estimation task, by assigning higher importance to more informative moments, and penalizing moments providing a small contribution for the reconstruction of the scene. As a result, in the formulation of our problem, such coefficients are assumed as parts of the inputs of the considered system, as described in the next paragraph.

4.3.1 System dynamics

Specifically, a typical SfM problem is formulated by considering the state vector $\boldsymbol{x} = (\boldsymbol{x}_m^T \ \boldsymbol{x}_u^T)^T$, where $\boldsymbol{x}_m$ and $\boldsymbol{x}_u$ represent the *measurable* and *unmeasurable* components of the state, respectively. The dynamics of $\boldsymbol{x}$ is given by

$$\begin{cases} \dot{\boldsymbol{x}}_m &= \boldsymbol{f}_m(\boldsymbol{x}_m, \boldsymbol{x}_u, \boldsymbol{u}, t) + \boldsymbol{\Omega}^T(t)\boldsymbol{x}_u \\ \dot{\boldsymbol{x}}_u &= \boldsymbol{f}_u(\boldsymbol{x}_m, \boldsymbol{x}_u, \boldsymbol{u}, t) \end{cases} \tag{4.11}$$

where $\boldsymbol{u}$ is the input of the system, $\boldsymbol{\Omega}$ is a time-varying known matrix and $\boldsymbol{f}_m$ and $\boldsymbol{f}_u$ are sufficiently smooth functions. The matrix $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$ is referred as *observability matrix* and plays an important role for the satisfaction of the estimation task. Indeed,

the full rankness of $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$ is a possible metrics to evaluate the Observability Rank Condition (ORC), thus it is a necessary and sufficient condition for the convergence of the estimation error.

The measurable component represents the visual measurements taken into account for the problem. In our case, we consider three linear combinations of photometric moments (4.10), by considering three different parameters vectors $\boldsymbol{\theta}_1$, $\boldsymbol{\theta_2}$ and $\boldsymbol{\theta}_3$, i.e.,

$$\boldsymbol{x}_m = \begin{pmatrix} m_w(\boldsymbol{\theta}_1) \\ m_w(\boldsymbol{\theta}_2) \\ m_w(\boldsymbol{\theta}_3) \end{pmatrix} = \begin{pmatrix} \boldsymbol{\theta}_1^T \\ \boldsymbol{\theta}_2^T \\ \boldsymbol{\theta}_3^T \end{pmatrix} \boldsymbol{m}_{p,q} = \boldsymbol{\Theta}^T \boldsymbol{m}_{p,q} \tag{4.12}$$

The unmeasurable component $\boldsymbol{x}_u$ encodes the geometric information that is intended to be estimated. In our case, we consider

$$\boldsymbol{x}_u = -\boldsymbol{n}/d = (A, B, C)^T \tag{4.13}$$

Differentiating (4.12) and (4.13) with respect to time, we get

$$\begin{aligned} \dot{\boldsymbol{x}}_m &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\boldsymbol{\Theta}^T \boldsymbol{m}_{p,q}\right) = \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\Theta}^T \boldsymbol{m}_{p,q} + \boldsymbol{\Theta}^T \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{m}_{p,q} = \\ &= \dot{\boldsymbol{\Theta}}^T \boldsymbol{m}_{p,q} + \boldsymbol{\Theta}^T \dot{\boldsymbol{m}}_{p,q} = \\ &= \dot{\boldsymbol{\Theta}}^T \boldsymbol{m}_{p,q} + \boldsymbol{\Theta}^T \bar{\boldsymbol{L}}_{m_{p,q}} \mathrm{v} \\ \\ \dot{\boldsymbol{x}}_u &= \boldsymbol{x}_u \boldsymbol{x}_u^T \boldsymbol{v} - \lfloor \boldsymbol{\omega} \rfloor_\times \boldsymbol{x}_u \end{aligned} \tag{4.14}$$

where $\lfloor * \rfloor_\times$ denotes the skew-symmetric matrix of its argument, and $\bar{\boldsymbol{L}}_{m_{p,q}} = \left(\boldsymbol{L}_{m_{00}}, \boldsymbol{L}_{m_{01}}, \ldots, \boldsymbol{L}_{m_{p,q}}\right)^T$ is built by stacking the interaction matrices defined in (4.8) for each moment in $\boldsymbol{m}_{p,q}$. Separating the contributions of linear and angular velocity in $\bar{\boldsymbol{L}}_{m_{p,q}}$, we write

$$\begin{aligned} \dot{\boldsymbol{x}}_m &= \dot{\boldsymbol{\Theta}}^T \boldsymbol{m}_{p,q} + \boldsymbol{\Theta}^T \left(\bar{\boldsymbol{L}}^{\boldsymbol{v}}_{m_{p,q}}(\boldsymbol{x}_u)\boldsymbol{v} + \bar{\boldsymbol{L}}^{\boldsymbol{\omega}}_{m_{p,q}}\boldsymbol{\omega}\right) = \\ &= \dot{\boldsymbol{\Theta}}^T \boldsymbol{m}_{p,q} + \boldsymbol{\Theta}^T \left(\bar{\boldsymbol{L}}^{\boldsymbol{x}_u}_{m_{p,q}}(\boldsymbol{v})\boldsymbol{x}_u + \bar{\boldsymbol{L}}^{\boldsymbol{\omega}}_{m_{p,q}}\boldsymbol{\omega}\right) = \\ &= \dot{\boldsymbol{\Theta}}^T \boldsymbol{m}_{p,q} + \boldsymbol{\Theta}^T \bar{\boldsymbol{L}}^{\boldsymbol{\omega}}_{m_{p,q}}\boldsymbol{\omega} + \boldsymbol{\Theta}^T \bar{\boldsymbol{L}}^{\boldsymbol{x}_u}_{m_{p,q}}(\boldsymbol{v})\boldsymbol{x}_u \end{aligned} \tag{4.15}$$

where we exploited the linearity of the dynamics (4.9) with respect to the plane parameters, i.e., $\bar{\boldsymbol{L}}^{\boldsymbol{x}_u}_{m_{p,q}}(\boldsymbol{v})\boldsymbol{x}_u = \bar{\boldsymbol{L}}^{\boldsymbol{v}}_{m_{p,q}}(\boldsymbol{x}_u)\boldsymbol{v}$. Comparing the expressions above with (4.11), we get the following correspondences

$$\begin{aligned} \boldsymbol{f}_m(\boldsymbol{x}_m, \boldsymbol{x}_u, \boldsymbol{u}, t) &= \dot{\boldsymbol{\Theta}}^T \boldsymbol{m}_{p,q} + \boldsymbol{\Theta}^T \bar{\boldsymbol{L}}^{\boldsymbol{\omega}}_{m_{p,q}}\boldsymbol{\omega} \\ \\ \boldsymbol{f}_u(\boldsymbol{x}_m, \boldsymbol{x}_u, \boldsymbol{u}, t) &= \boldsymbol{x}_u \boldsymbol{x}_u^T \boldsymbol{v} - \lfloor \boldsymbol{\omega} \rfloor_\times \boldsymbol{x}_u \\ \\ \boldsymbol{\Omega} &= \bar{\boldsymbol{L}}^{\boldsymbol{x}_u\ T}_{m_{p,q}}(\boldsymbol{v})\boldsymbol{\Theta} \\ \\ \boldsymbol{u} &= (\mathrm{v}^T, \boldsymbol{\theta}^T)^T \end{aligned} \tag{4.16}$$

4.3.2 Observer dynamics

The goal of the proposed strategy is to estimate the unmeasurable component $\boldsymbol{x}_u$, while satisfying some optimality criteria on the performance of the estimation. Specifically, we want to generate the optimal camera velocity $\boldsymbol{v}$ and the vector of parameters $\boldsymbol{\theta}$ to maximize a given conditioning measurement of $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$, so as to minimize the convergence rate of the designed observer. We highlight that, given the form of $\boldsymbol{\Omega}$ in (4.16), the performances of the structure reconstruction and the estimation process depend only on the linear velocity $\boldsymbol{v}$. This is a typical aspect of the SfM schemes, and allows to consider the camera angular velocity $\boldsymbol{\omega}$ as additional degrees of freedom to satisfy other objectives (e.g., achieving a given control task).

To reconstruct the unmeasurable component $\boldsymbol{x}_u$, a possible non-linear observer scheme is given by

$$\left\{ \begin{array}{rcl} \dot{\hat{\boldsymbol{x}}}_m & = & \boldsymbol{f}_m(\boldsymbol{x}_m, \boldsymbol{x}_u, \boldsymbol{u}, t) + \boldsymbol{\Omega}^T(t)\boldsymbol{x}_u + \boldsymbol{H}\boldsymbol{\xi} \\ \dot{\hat{\boldsymbol{x}}}_u & = & \boldsymbol{f}_u(\boldsymbol{x}_m, \hat{\boldsymbol{x}}_u, \boldsymbol{u}, t) + \boldsymbol{\Lambda}\boldsymbol{\Omega}(t)\boldsymbol{P}\boldsymbol{\xi} \end{array} \right. \tag{4.17}$$

where $\boldsymbol{H} > 0$, $\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^T > 0$ and $\boldsymbol{P} = \boldsymbol{P}^T > 0$ are tuning parameters of the non-linear observer, so one can act on them to satisfy specific conditions. Without loss of generality, we consider $\boldsymbol{P} = \alpha\boldsymbol{I}$ and $\boldsymbol{\Lambda} = \beta\boldsymbol{I}$, $\alpha > 0$, $\beta > 0$. Denoting by $\boldsymbol{\xi} = \boldsymbol{x}_m - \hat{\boldsymbol{x}}_m$ and $\boldsymbol{z} = \boldsymbol{x}_u - \hat{\boldsymbol{x}}_u$ the *measurable* and the *unmeasurable* errors, respectively, the dynamics of the estimation error $\boldsymbol{e} = [\boldsymbol{\xi}, \boldsymbol{z}]^T$is given by

$$\left\{ \begin{array}{rcl} \dot{\boldsymbol{\xi}} & = & -\boldsymbol{H}\boldsymbol{\xi} + \boldsymbol{\Omega}^T\boldsymbol{z} \\ \dot{\boldsymbol{z}} & = & -\boldsymbol{\Lambda}\boldsymbol{\Omega}\boldsymbol{P}\boldsymbol{\xi} + (\boldsymbol{f}_m(\boldsymbol{x}_m, \boldsymbol{x}_u, \boldsymbol{u}, t) - \boldsymbol{f}_m(\boldsymbol{x}_m, \hat{\boldsymbol{x}}_u, \boldsymbol{u}, t)) = \\ & = & -\boldsymbol{\Lambda}\boldsymbol{\Omega}\boldsymbol{P}\boldsymbol{\xi} + \boldsymbol{g}(\boldsymbol{e}, t) \end{array} \right. \tag{4.18}$$

where $\boldsymbol{g}(\boldsymbol{e}, t)$ is a *vanishing* perturbation, since $\boldsymbol{g}(\boldsymbol{0}, t) = \boldsymbol{0}$, $\forall t$. In particular, it is shown that matrices $\boldsymbol{\Lambda}$ and $\boldsymbol{P}$ are tuned to attenuate $\boldsymbol{g}(\boldsymbol{e}, t)$, while keeping the norm of $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$ limited. On the other hand, the matrix $\boldsymbol{H}$ is tuned to shape the damping factor of the error during the transient. Specifically, $\boldsymbol{H}$ is chosen as

$$\boldsymbol{H} = \boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^T \tag{4.19}$$

where $\boldsymbol{V}$ is the matrix whose columns are the singular vectors of $\boldsymbol{\Omega}$, according to the SVD decomposition $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T = \boldsymbol{\Omega}$, and $\boldsymbol{D} = diag(2\sigma_i\sqrt{\alpha\beta})$, being σ_i the singular values of $\boldsymbol{\Omega}$. To optimize the system inputs and minimize the convergence rate of the estimation, we consider to maximize the determinant of the observability matrix:

$$\rho = \det(\boldsymbol{\Omega}\boldsymbol{\Omega}^T) \tag{4.20}$$

By differentiating (4.20) with respect to time, we end up with the following relationship

$$\dot{\rho} = \boldsymbol{J}_{\boldsymbol{v}}\dot{\boldsymbol{v}} + \boldsymbol{J}_{\boldsymbol{\theta}}\dot{\boldsymbol{\theta}} + \boldsymbol{J}_{\boldsymbol{m}}\dot{\boldsymbol{m}}_{p,q} \tag{4.21}$$

where $\boldsymbol{J}_{\boldsymbol{v}}$, $\boldsymbol{J}_{\boldsymbol{\theta}}$, $\boldsymbol{J}_{\boldsymbol{m}}$ are Jacobian matrices that can be computed in closed-form from $\boldsymbol{\Omega}$. The expression (4.21) states that, to affect and maximize the determinant ρ of the matrix $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$, we can drive the system inputs $\dot{\boldsymbol{v}}$ and $\dot{\boldsymbol{\theta}}$ towards the directions

given by $\boldsymbol{J_v}$ and $\boldsymbol{J_\theta}$. Specifically, they are computed as

$$\begin{aligned} \boldsymbol{J_v} &= \left[\dots \operatorname{tr}\left(\operatorname{adj}\left(\boldsymbol{\Omega\Omega}^T\right) \frac{\partial \boldsymbol{\Omega\Omega}^T}{\partial v_i} \right) \dots \right] \\ \boldsymbol{J_\theta} &= \left[\dots \operatorname{tr}\left(\operatorname{adj}\left(\boldsymbol{\Omega\Omega}^T\right) \frac{\partial \boldsymbol{\Omega\Omega}^T}{\partial \theta_i} \right) \dots \right] \end{aligned} \tag{4.22}$$

where, given the form of $\boldsymbol{\Omega}$ in (4.16), we can compute the terms

$$\begin{aligned} \frac{\partial \boldsymbol{\Omega\Omega}^T}{\partial v_i} &= \frac{\partial \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u\ T}(\boldsymbol{v})}{\partial v_i} \boldsymbol{\Theta\Theta}^T \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u}(\boldsymbol{v}) + \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u\ T}(\boldsymbol{v}) \boldsymbol{\Theta\Theta}^T \frac{\partial \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u}(\boldsymbol{v})}{\partial v_i} \\ &= \boldsymbol{M} + \boldsymbol{M}^T \\ \frac{\partial \boldsymbol{\Omega\Omega}^T}{\partial \theta_i} &= \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u\ T}(\boldsymbol{v}) \frac{\partial \boldsymbol{\Theta}}{\partial \theta_i} \boldsymbol{\Theta}^T \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u}(\boldsymbol{v}) + \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u\ T}(\boldsymbol{v}) \boldsymbol{\Theta} \frac{\partial \boldsymbol{\Theta}}{\partial \theta_i}^T \bar{\boldsymbol{L}}_{m_{p,q}}^{\boldsymbol{x}_u}(\boldsymbol{v}) \\ &= \boldsymbol{N} + \boldsymbol{N}^T \end{aligned} \tag{4.23}$$

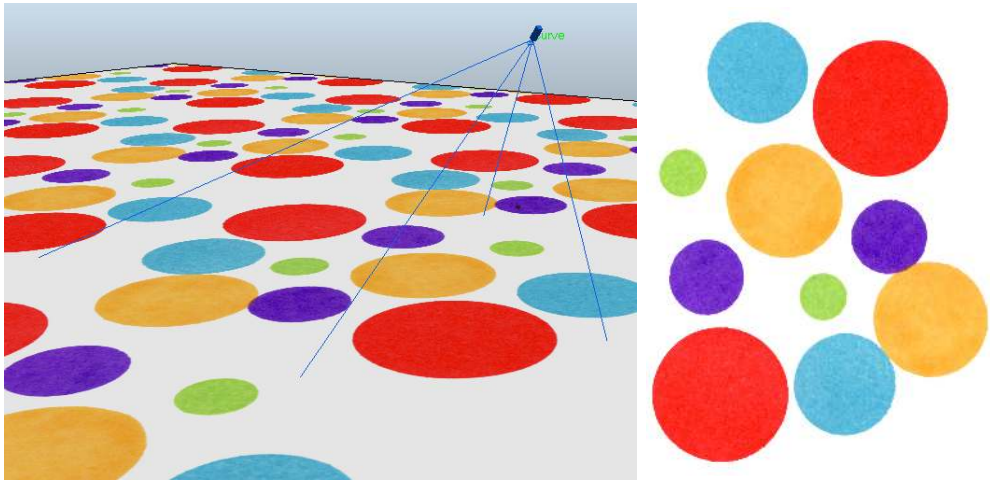
From (4.16), since the norm of $\boldsymbol{\Omega}$ (thus, the determinant) is related to the norm of $\boldsymbol{v}$, to avoid that $\boldsymbol{\Omega\Omega}^T$ is increased by only increasing $\boldsymbol{v}$, for the generation of the desired inputs we consider to keep $\|\boldsymbol{v}\| = const$ as additional constraint. Similarly, we also consider to keep $\|\boldsymbol{\theta}\|$ to prevent an undesired increase of the noise in the maximization of ρ. In detail, defining $\kappa_v = \frac{1}{2}\|\boldsymbol{v}\|^2$, $\kappa_\theta = \frac{1}{2}\|\boldsymbol{\theta}\|^2$, and denoting by $\kappa_{v,d}$ and $\kappa_{\theta,d}$ the corresponding desired values, to generate the proper inputs that maximize ρ, while keeping the norm of the input quantities constant, we finally design the following update rules

$$\begin{cases} \dot{\boldsymbol{v}} = \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|^2} K_{v,1}(\kappa_{v,d} - \kappa_v) + K_{v,2} \left(\boldsymbol{I} - \frac{\boldsymbol{v}\boldsymbol{v}^T}{\|\boldsymbol{v}\|^2} \right) \boldsymbol{J}_{\boldsymbol{v}}^T \\ \dot{\boldsymbol{\theta}} = \frac{\boldsymbol{\theta}}{\|\boldsymbol{\theta}\|^2} K_{\theta,1}(\kappa_{\theta,d} - \kappa_\theta) + K_{\theta,2} \left(\boldsymbol{I} - \frac{\boldsymbol{\theta}\boldsymbol{\theta}^T}{\|\boldsymbol{\theta}\|^2} \right) \boldsymbol{J}_{\boldsymbol{\theta}}^T \end{cases} \tag{4.24}$$

The optimization rules expressed above maximize the determinant ρ as observability measure of the observability matrix $\boldsymbol{\Omega\Omega}^T$ of the SfM task. This is done by acting both on the camera linear velocity $\boldsymbol{v}$ and on vector of parameters $\boldsymbol{\theta}$ representing the coefficients of the linear combination of photometric moments. This way, the most informative image moments are adaptively selected to optimize the observability conditions of the considered estimation task.

4.4 Simulation results

To validate the presented scheme, we considered a simulated scenario where a free-flying camera sensor moves over a planar scene, as depicted in Figure 4.2a. A simple texture with colored circles has been considered to provide informative data to the observed plane (Figure 4.2b).



(a) (b)

Figure 4.2. (a) A free-flying camera sensor moving over a planar scene in the V-REP simulation environment. (b) Texture adopted for the plane.

For the considered simulation, we set $p = q = 2$ as the maximum order of image moments to be propagated and considered in the vector $\boldsymbol{m}_{p,q}$. At time $t_0 = 0$ we considered $\boldsymbol{\xi}_0 = \mathbf{0}$ and $\boldsymbol{z}_0 = [0.5,\ 1,\ -0.5]^T$, with initial camera velocity $\boldsymbol{v}_0 = [-0.1,\ 0.1,\ -0.1]^T$ and $\boldsymbol{\theta}_0$ randomly selected in the range $[-1,\ 1]$. The matrices $\boldsymbol{\Lambda}$ and $\boldsymbol{P}$ are set such that $\alpha\beta = 1000$, while $K_{v,1} = K_{\theta,1} = 1.0$, $K_{v,2} = 3.0$ and $K_{\theta,2} = 7.0$. We ran two different simulations: in the first case, we only optimized the camera velocity $\boldsymbol{v}$, while leaving the parameters $\boldsymbol{\theta}$ constant over time during the simulation (i.e., setting $K_{\theta,2} = 0$). In the second case, we considered the full optimization scheme by also considering the maximization of ρ with respect to the system inputs $\dot{\boldsymbol{\theta}}$. Figure 4.3 shows related plots of a comparison between the two considered scenarios. It is noticeable how the optimization of the $\boldsymbol{\theta}$ coefficients strongly affects the overall performance of the estimation process: Figure 4.3a shows that, when the $\boldsymbol{\theta}$ are optimized, the determinant ρ of $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$ reaches a higher value with a peak of $\approx 1.7 \cdot 10^{-4}$, compared to the case in which the $\boldsymbol{\theta}$ are left unchanged, where a value of $\approx 2.5 \cdot 10^{-6}$ is reached as maximum. This has effects on the convergence rate of the estimation process. Figure 4.3b highlights that the optimization of both $\boldsymbol{v}$ and $\boldsymbol{\theta}$ allows to achieve quick convergence of the $\boldsymbol{z}$ error in about $8s$, while the only optimization of $\boldsymbol{v}$ is not sufficient to achieve smooth convergence before $14s$. This can be also appreciated in terms of error on the components $\boldsymbol{n}$ and d describing the equation of the plane. Specifically, recalling that $\boldsymbol{n}$ and d are retrieved from $\boldsymbol{x}_u$ as $\boldsymbol{n} = -\frac{\boldsymbol{x}_u}{\|\boldsymbol{x}_u\|}$ and $d = 1/\|\boldsymbol{x}_u\|$, we evaluate the errors $e_d = d - \hat{d}$ and $e_{\boldsymbol{n}} = \arccos(\boldsymbol{n}^T\hat{\boldsymbol{n}})$, whose evaluations are shown in Figure 4.3c 4.3d. After convergence, the estimation error on the d component is around ≈ 10cm, while the error on the $\boldsymbol{n}$ component is $\approx 5°$.

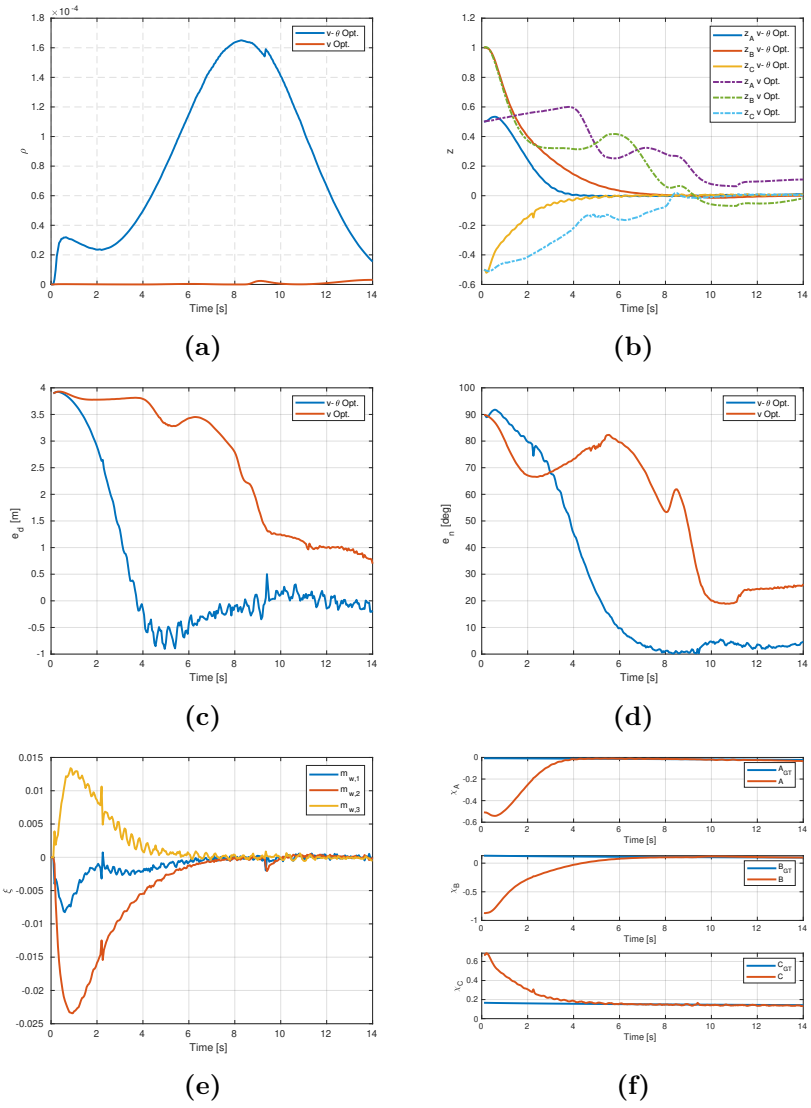


Figure 4.3. Plots related to the simulations performed to validate the active SfM scheme, with comparison between the case in which both $\boldsymbol{v}$ and $\boldsymbol{\theta}$ are optimized, and the case in which $\boldsymbol{\theta}$ are left unchanged. (a) Determinant ρ of the observability matrix $\boldsymbol{\Omega}\boldsymbol{\Omega}^T$. (b) Unmeasurable error $\boldsymbol{z}$. (c) Unit normal vector error $e_{\boldsymbol{n}}$. (d) Distance error e_d. (e) Measurable component error $\boldsymbol{\xi}$ in case of full optimization. (f) Convergence of the estimated plane coefficients $\boldsymbol{x}_u$ towards the ground truth values, in case of full optimization.

4.5 Conclusions

In this chapter, we addressed the problem of active perception for a Strucure-from-Motion problem, in case of a planar scene observed by a moving camera. For the purpose, an active framework has been adapted to handle dense visual information in terms of photometric image moments. This prevented to employ critical image processing steps that could introduce undesired noise in the measurement process. In addition, a weighted expression of moments also allowed to manage discontinuities

due to appearance/disappearance of content image from the camera plane. Such moments have been combined by generating a weighted sum whose coefficients are adaptively changed to select the best moment set in terms of information gain for the estimation process. Specifically, the determinant of the observability matrix associated to the problem has been chosen as metrics to maximize, in order to minimize the convergence rate of the estimation. Some preliminary results have been presented in simulated scenario, showing the validity and the effectiveness of the approach. Future works will consider a fair comparison with well-known and widely-used set of moments, to compare the effects of the active coefficients with respect to feature set established a priori. Furthermore, an experimental session on a robot manipulator arm is planned, in order to make the validation of the presented method more robust.

Chapter 5

Conclusions

This manuscript presented vision-based methods for control and state estimation of robotic systems. First, a safe navigation scheme for mobile robots, moving in unknown environments populated by obstacles, has been presented. For the purpose, dense visual information has been exploited to perceive the environment (i.e., detect ground plane and obstacles) and, in combination with other sensory sources, provide an estimation of the robot motion with a linear observer. On the other hand, sparse visual data have been extracted in terms of geometric primitives, in order to propose several visual servoing control schemes satisfying proper navigation behaviors. A comparative discussion has been proposed, and finally a controller with velocity feedback, relied on visual estimated information, has been designed to guarantee safety during navigation.

For surgical robotics, we presented a Kalman-based observer to estimate the 3D pose of a suturing needle held by a surgical manipulator for robot-assisted suturing. The method exploited images acquired by the endoscope of the robot platform to extrapolate relevant geometrical information and get projected measurements of the tool pose. This method has also been validated with a novel simulator designed for the da Vinci robotic platform, with the purpose to ease interfacing and employment in ideal conditions for testing and validation.

Finally, an *active estimation* paradigm has been introduced. In particular, a novel active sensing algorithm employing visual dense information has been described for a typical Structure-from-Motion (SfM) problem. The algorithm generated an optimal estimation of a scene observed by a moving camera, while minimizing the maximum uncertainty of the estimation. This approach can be applied to any robotic platforms, and has been validated in simulation with a free-flying camera.

Appendix A

V-REP da Vinci Simulator

The da Vinci Research Kit (dVRK) [71] is an open-source mechatronic platform obtained from the first-generation of Intuitive Surgical System and provided with controllers and software developed at Johns Hopkins University LCSR and Worcester Polytechnic Institute AIM Lab [92]. The research community sharing the dVRK is composed by over 30 research institutions across the world, which provides an idea of the importance of this open platform in technology-oriented research for robot-aided surgery. By default the daVinci is a master-slave robot scaling the motion, attenuating tremor and enhancing precision of the surgeon that remotely control the system. Despite teleoperation is the primary control mode, sensor-based shared control of robot trajectories is under development to augment surgeon abilities [93]. Thus, the open research platform is suitable to enhance both research in haptic teleoperation [94] and in semi-autonomus control [95]. In the last two decades, surgical simulation has strongly widespread thanks to the progress in robotic surgery [96]. Simulation and virtual reality support different research fields from industry to entertainment up to surgical robotics. In the surgical field, simulators are mainly developed for training to allow surgeons acquiring basic robotic skills as well as more complex maneuvers before performing live surgery. In the face of cost/effectiveness ratio, simulators are a solution to allow students learning the base techniques for robot-assisted surgery. The state of the art of simulators currently available is constituted by: *Robotic Surgery Simulator (RoSS)* [97], *SimSurgery Education Platform (SEP)* [98], *da Vinci Trainer* [99], *da Vinci Skills Simulator* [100], *Robotix Mentor* [101] and *Chiron* [102]. In [103] a comparative evaluation of some of these simulators is provided to help users in selecting an appropriate device for their needs. Besides the training capabilities of each system, all of them provide modules for EndoWrist manipulation, camera control, needle control and clutching, and a realistic representation of the da Vinci workspace. The described simulators are designed specifically for surgeons' training and do not provide a virtual reality simulator of the whole robot kinematics, namely Setup joints (SUJ), Patient Side Manipulators (PSMs), Endoscopic Camera Manipulator (ECM). On the other hand, simulating the robot is a solution providing a low cost and easy to access environment for the development and proof of new control strategies while minimizing the risk of testing new algorithms on such a complex system. Furthermore, it is a valuable tool for safely testing out new technology, e.g. new surgical tools and sensors [104, 105].

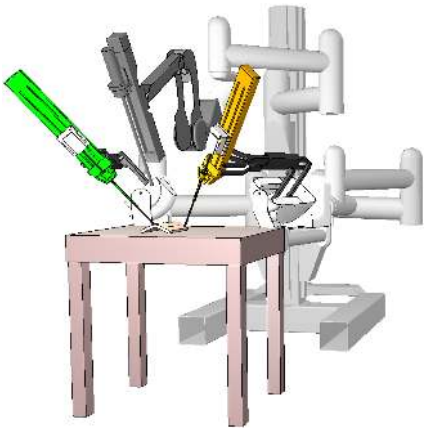


Figure A.1. The da Vinci Research Kit V-REP simulator.

In a wide community sharing the dVRK a free simulator is a benefit for the progress of the research whereas the uniqueness of the robot, received as a donation from Intuitive Surgical, and the difficulty of replacing components in case of malfunctioning. Furthermore, a simulator will also profit educational purposes as it enables students to easily approach the system and work in total safety. A brief description of the most advanced open source software packages for robotic simulation can be found in [106]. The use of open source robot simulators allows creating a virtual model of a robot, simulating components and sensors and testing new tools design, control strategies and integrating learning in a simulation environments. In this paper, the dVRK simulator is realized using V-REP [107]. The choice is motivated by the versatility and simplicity of this software for multi-robot applications. V-REP is based on a distributed control architecture. Each object/model can be individually controlled via an embedded script, a plugin, a ROS or BlueZero node, a remote API client, or a custom solution. Controllers can be written in C/C++, Python, Java, Lua, Matlab or Octave. Therefore, the simulator can be easily interfaced with the real surgeon master console, and new objects and robots can be imported in the scene by using a graphical interface. The developed simulator includes the kinematics of the SUJ, PSMs, ECM and the camera sensor and it is interfaced with the ROS framework. Moreover, four scenes are already created and ready for use.

The complete simulator, together with the four developed application scenes, is available at `https://github.com/unina-icaros/dvrk-vrep.git`.

The rest of this appendix is organized as follows: first, we describe the kinematics of the robotic arms included in the simulator; then, the V-REP models, focusing on the simulated scene and on the simulator performances, and the control architecture, focusing on the ROS-based infrastructure, are reported. After the description of the simulator, we also present an extension to reproduce the Master surgeon console, through the employment of a pair of haptic devices and a virtual reality head-set display.

A.1 da Vinci Simulator

A.1.1 The da Vinci surgical system kinematic model

The full dVRK is a decommissioned first generation da Vinci Surgical System consisting of two/three PSMs, one ECM, and two MTMs. All the slave side manipulators are mounted on a SUJs that allows the manual spatial positioning of the arm bases. We include in the simulated environment all the patient side manipulators composed of two PSMs and an ECM mounted on the SUJ. In the next sections, a brief description of the arms kinematics is reported.

Setup Joint arm kinematics

The two PSMs and the ECM are mounted on an articulated robotic structure composed by three or, in the newest versions, four arms SUJs[1]. The two PSMs are located at the end of two 6-degree-of-freedom (DoFs) arms (that we indicate hereafter as SUJ-PSMs) while the ECM is located at the end of a 4-DoFs arm (SUJ-ECM). All the robotic arms in the SUJs are not actuated by motors but it is possible to control breaks in each joint and read the angular position using potentiometers [71]. Denoting with $\boldsymbol{q}_{sp} = [q_{sp,1}, \dots, q_{sp,6}]$ the vector of the SUJ-PSMs arms generalized coordinates, the homogeneous transformation matrix[2] $\boldsymbol{T}^{\mathcal{B}}_{\mathcal{AP}}(\boldsymbol{q}_{sp}) \in SE(3)$, representing the pose of the SUJ-PSMs end effector frame $\mathcal{AP} : \{\boldsymbol{O}_{ap}; \boldsymbol{x}_{ap}, \boldsymbol{y}_{ap}, \boldsymbol{z}_{ap}\}$ with respect to the base frame $\mathcal{B} : \{\boldsymbol{O}_b; \boldsymbol{x}_b, \boldsymbol{y}_b, \boldsymbol{z}_b\}$, can be easily computed applying the standard DH convention to the kinematic chain $\{J_1, \dots, J_6\}$ of Fig. A.2 (see Table A.1 where $a_2 = 0.58$m, $a_3 = 0.56$m and $d_4 = 0.425$m). Moreover, denoting with $\boldsymbol{q}_{se} = [q_{se,1}, \dots, q_{se,4}]$ the vector of the SUJ-ECM arm generalized coordinates, the pose of the SUJ-ECM end effector frame $\mathcal{AE} : \{\boldsymbol{O}_{ae}; \boldsymbol{x}_{ae}, \boldsymbol{y}_{ae}, \boldsymbol{z}_{ae}\}$ with respect to the base frame $\mathcal{B} : \{\boldsymbol{O}_b; \boldsymbol{x}_b, \boldsymbol{y}_b, \boldsymbol{z}_b\}$, defined by the homogeneous transformation matrix $\boldsymbol{T}^{\mathcal{B}}_{\mathcal{AE}}(\boldsymbol{q}_{se}) \in SE(3)$, can be computed considering only the first four rows in Table A.1. Notice that, two constant homogeneous transformation matrix $\boldsymbol{T}^{\mathcal{AP}}_{\mathcal{BP}} \in SE(3)$ and $\boldsymbol{T}^{\mathcal{AE}}_{\mathcal{BE}} \in SE(3)$ must be considered to complete che kinematics description and link $\mathcal{AP}$ and $\mathcal{AE}$ (respectively the last SUJ-PSM and SUJ-ECM frames) to the base frames $\mathcal{BP}$ and $\mathcal{BE}$ of the PSMs and of the ECM described in Sec. A.1.1 and A.1.1, respectively (see Fig. A.2).

Table A.1. DH parameters of the SUJ

link	joint	a_i	α_i	d_i	θ_i
1	P	0	0	$q_{se,1}$	–
2	R	a_2	0	–	$q_{se,2}$
3	R	a_3	0	–	$q_{se,3}$
4	R	0	$-\pi/2$	–	$q_{se,4}$
5	R	0	$\pi/2$	$-d_4$	$q_{se,5}$
6	R	0	0	–	$q_{se,6}$

[1] In this work we consider the three arm version of the SUJs.

[2] Hereafter, we use the matrix notation $\boldsymbol{T}^a_b$, where the superscript a denotes the frame in which vector components are expressed, the subscript b the current frame. *E.g.*, $\boldsymbol{T}^{\mathcal{B}}_{\mathcal{AP}}$ denotes the pose of the SUJ-PSM attach point expressed in the base frame.

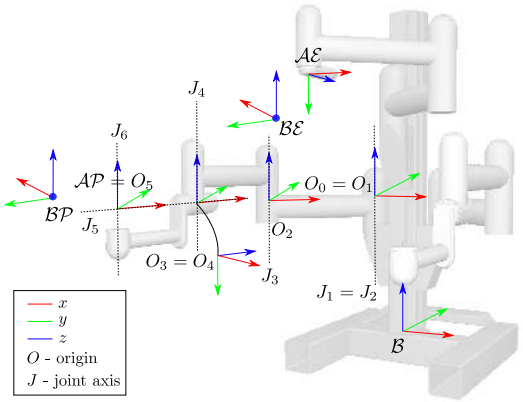


Figure A.2. SUJs kinematic description.

PSM arm kinematics

Each PSM is a 7-DoF actuated arm, which moves a surgical instrument about a Remote Center of Motion (RCM), *i.e.*, a fixed fulcrum point that is invariant to the configuration of the PSM joints [108, 109]. The first 6 DoFs correspond to Revolute (R) or Prismatic (P) joints, combined in a RRPRRR sequence. The last DoF corresponds to the opening and closing motion of the gripper. The homogeneous transformation matrix $\boldsymbol{T}_{\mathcal{G}}^{\mathcal{BP}}(\boldsymbol{q}_p) \in SE(3)$ (where $\boldsymbol{q}_p = [q_{p,1}, \ldots, q_{p,6}]$ indicates the vector of the PSM generalized coordinates), representing the pose of the gripper frame $\mathcal{G} : \{\boldsymbol{O}_g; \boldsymbol{x}_g, \boldsymbol{y}_g, \boldsymbol{z}_g\}$ with respect to the base frame $\mathcal{BP} : \{\boldsymbol{O}_{bp}; \boldsymbol{x}_{bp}, \boldsymbol{y}_{bp}, \boldsymbol{z}_{bp}\}$, can be easily computed by choosing the origin of frame $\mathcal{BP}$ in the RCM point and applying the standard DH convention to the kinematic chain $\{J_1, \ldots, J_6\}$ of Fig. A.3b (see Table A.2, where $a_5 = 0.0091$ m).

Table A.2. DH parameters of the PSM

link	joint	a_i	α_i	d_i	θ_i
1	R	0	$-\pi/2$	–	$q_{p,1}$
2	R	0	$-\pi/2$	–	$q_{p,2}$
3	P	0	0	$q_{p,3}$	–
4	R	0	$\pi/2$	–	$q_{p,4}$
5	R	a_5	$-\pi/2$	–	$q_{p,5}$
6	R	0	$-\pi/2$	–	$q_{p,6}$

ECM arm kinematics

The ECM is a 4-DoF actuated arm, which moves the endoscopic camera about the RCM through revolute and prismatic joints, combined in a RRPR sequence. The homogeneous transformation matrix $\boldsymbol{T}_{\mathcal{C}}^{\mathcal{BC}}(\boldsymbol{q}_e) \in SE(3)$ (with the vector $\boldsymbol{q}_e = [q_{e,1}, \ldots, q_{e,4}]$), representing the pose of the camera frame $\mathcal{C} = \{\boldsymbol{O}_c; \boldsymbol{x}_c, \boldsymbol{y}_c, \boldsymbol{z}_c\}$ with respect to the base frame $\mathcal{BC} = \{\boldsymbol{O}_{bc}; \boldsymbol{x}_{bc}, \boldsymbol{y}_{bc}, \boldsymbol{z}_{bc}\}$, can be easily computed by choosing the origin of frame $\mathcal{CB}$ in the RCM point and applying the standard DH

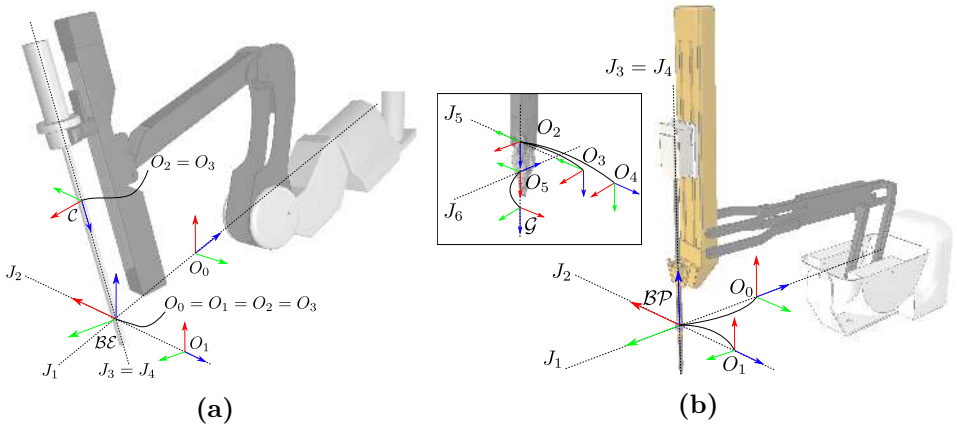


Figure A.3. (a) ECM kinematic description. (b) PSM kinematic description.

convention to the kinematic chain $\{J_1, \ldots, J_4\}$ of Fig. A.3a (parameters are given in Table A.3, where $d_4 = 0.007$m).

Table A.3. DH parameters of the ECM

link	joint	a_i	α_i	d_i	θ_i
1	R	0	$-\pi/2$	–	$q_{e,1}$
2	R	0	$-\pi/2$	–	$q_{e,2}$
3	P	0	0	$q_{e,3}$	–
4	R	0	0	d_4	$q_{e,4}$

The robotic arms have been modeled starting from the CAD models included in [110] except for the SUJs. We realized the kinematic chain of each robotic arm by linking mesh and joints in a *joint-respondable-visual* sequence. The dynamics parameters of the PSM links have been identified through the method described in [109]. At the end of the endoscope link, two cameras have been included to simulate the binocular vision system of the real dVRK endoscope. We set a resolution for the cameras at 320×288 pixels, *i.e.*, half the resolution of the real endoscope, that results a trade-off option to have a good resolution and a good simulated sampling time.

A.1.2 Control architecture

The presented V-REP simulator has been developed to be fully integrated into the dVRK control infrastructure. Therefore, we used the high-level ROS framework to link the simulator to the low-level control [92]. This allows the user to use the simulator in different modalities: (*i*) telemanipulated using the dVRK MTMs; (*ii*) in combination with the real robotic PSMs and ECM, to implement augmented reality algorithms; (*iii*) as standalone, by controlling the simulated robot using the ROS framework (*e.g.*, through C++, MATLAB and Python ROS nodes), or directly in V-REP using the embedded scripts.

With reference to Fig. A.4, the control architectures of the dVRK, described in detail in [111], is composed of: (*i*) a hardware interface to communicate through

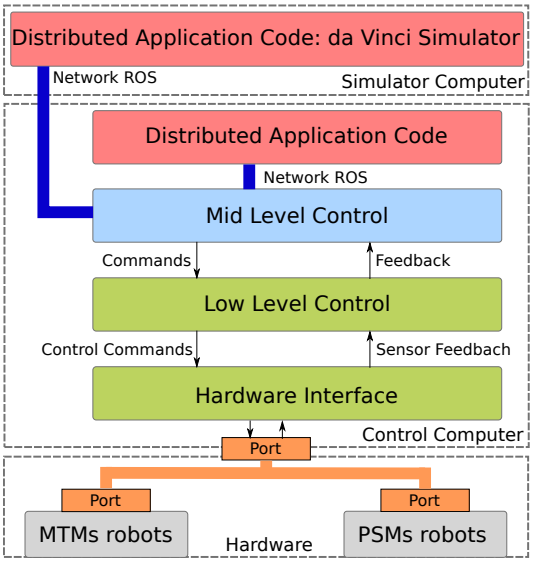


Figure A.4. Control architecture.

the fire-wire bus with the embedded actuator controllers and implementing the safety checks; (*ii*) a low level layer implementing all the algorithms for the inverse kinematics, impedance master control etc.; (*iii*) a mid-level layer implementing the ROS communication and the high level controllers. The communication between the da Vinci simulator, supposed to be running in a dedicate computer, and the dVRK console is implemented through ROS topics. In detail, we use the *v_repExtRosInterface* to publish the state of the robot joints (PSMs, SUJ, ECM) and the gripper state for the PSMs. Moreover, the simulator subscribes to two topics *sensor_msgs::joint_state* to control the robots joints motion from ROS.

Considering the computer configuration described in the previous section, all the joints and objects topics, included the cameras topics, are streamed at 60 Hz [3]. If the vision sensors in the scene are disabled, the joints and objects topics are streamed at 220Hz.

This architecture allows to easily interface the simulator with the mid level control of the dVRK (for commanding the simulated robot through MTMs) or to other ROS-integrated input device (*e.g.*, haptic devices).

In the next Section, we present a number of V-REP example scenes, describing a comprehensive set of possible applications that can be realized with our simulator, in order to highlight its functionalities.

A.1.3 Example scenes

The possibility to include different robots, dynamic objects, devices and sensors allows to easily extend the simulator capabilities through the creation of advanced V-REP scenes. In this work, we propose different scenes to show the potentialities addressing the implementation aspects, and representing common applications for robotic surgery research.

[3]The simulation requires to be run in *threaded-rendering* mode, in order to decouple the rendering and the control scripts and speed up the execution.

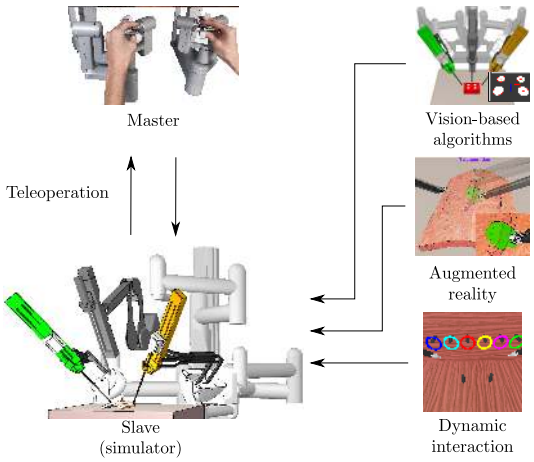


Figure A.5. Simulated environment.

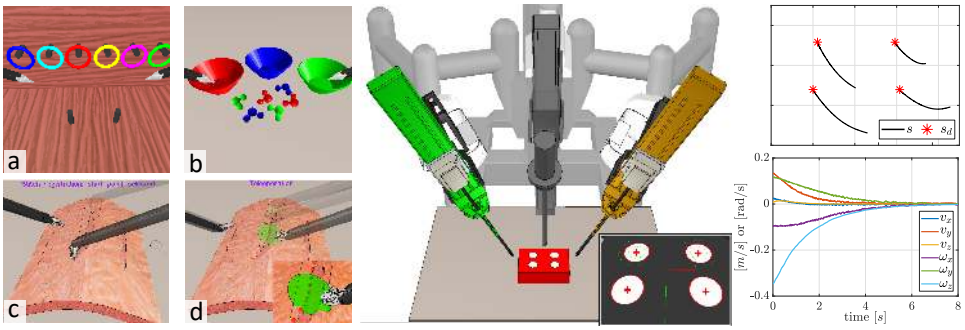
Hence, we show the development of advanced control strategies, *e.g.*, visual servoing or vision-based object tracking, augmented reality and simulation of rigid objects dynamics and interaction (see Fig. A.5).

In detail, we propose:

- Two training scenes, developed to show the the capability of the simulated robot interacting with rigid dynamic objects;
- A suturing scene, developed to show the possibility to easily integrate augmented reality information inside the simulated environment, and show an example of combination of autonomous and tele-operated task execution (assisted suturing);
- A needle tracking and a visual servoing scene, showing the potentiality of the environment to implement advanced vision-based algorithms, through the information acquired from the simulated vision system.

In order to effectively use the dVRK in surgical scenarios, surgeons spend a huge amount of time training in simulation. Intuitive Surgical provides surgical platforms and simulators embedding training modules for robotic skills, procedural tasks and complete robotic procedures. In this context, simulation is very important since it can provide scores information about the surgeon skills. However, these simulators are costly and not completely exploitable by roboticists, whose aim is to use the robot for research purposes. Novel engineers may equally need to train themselves to develop and test novel control strategies. To this end, we provide two V-REP scenes in which non-surgical training tasks are proposed, namely: *pick & place*, and *peg on board*. However, the high versatility of V-REP allows easy development and implementation of other training tasks and/or assistive strategies. Figure A.6 contains snapshots of the proposed scenes taken from the ECM left camera.

The scenes have been realized by developing and importing CAD models of the training setup into the scene. Embedded V-REP functions, allowed to create respondable and simplified dynamic entities used to simulate contacts and interaction among objects. Moreover, the control architecture presented in Sect. A.1.2 allows

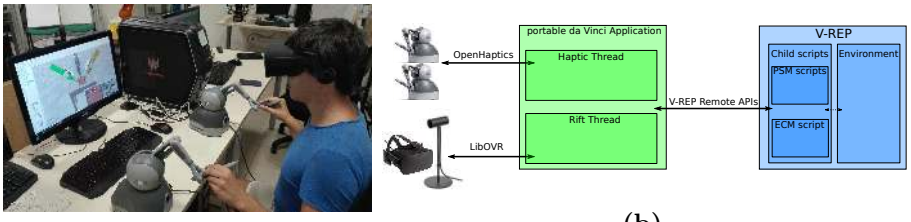


(a) **(b)**

Figure A.6. (a) Training and suturing scene setup. Upper row: two examples of training tasks, (a) peg on board; (b) pick and Place. Lower row: augmented reality suturing scene, (c) wound registration; (d) stitches planning and execution.(b) Visual servoing scene setup. Right-top: image plane features; right-bottom: camera velocities.

to easily interface the simulated robot with the MTMs or other input devices (*e.g.* haptic devices). Finally, a proximity sensor integrated between the needle driver pads has been used to simulate objects grasping.

A.2 Extension of the Simulator: a *portable* da Vinci



(a) **(b)**

Figure A.7. (a) The presented *portable* da Vinci simulated system. (b) Module and device communication scheme of the portable da Vinci application.

To fully reproduce the experience of controlling the da Vinci system, it is necessary to simulate also the Master console. Specifically, the Master console of the da Vinci has two main purposes: (i) sends commands to the PSMs through the pair of MTMs; (ii) shows the images acquired by the cameras mounted on the ECM, through a two-channel 3D vision system. To implement these two functionalities in our simulator, we consider a pair of haptic interfaces to simulate the pair of MTMs, while the 3D vision system is replicated through the use of a virtual reality headset. We highlight that, when using the console to control the patient-side manipulators, the 3D vision system displaying the images acquired by the endoscopic cameras is fixed, and the surgeon has no possibility to directly control the camera perspective view. By using a virtual reality headset in our setup, instead, we allow the user to freely move the head-mounted display (HMD) and directly control the ECM through the movements of his head. Along with providing a fully immersive experience,

this actually extends the potentials of the real robotic system, as the camera view changes in a more intuitive way.

In detail, we considered a pair of Geomagic Touch[4] haptic interfaces and the Oculus Rift[5] virtual reality headset, as devices required to faithfully reproduce the da Vinci Master console in a low-cost and easy-to-access fashion, thus realizing a *portable* da Vinci simulated system. Such functionalities are easily implemented through the native SDK of the devices, made available by the manufacturer companies to access the internal states of the devices. We use the *OpenHaptics* library to read data and command the Geomagic Touch devices, while the Oculus Rift functionalities are implemented through the *Oculus PC SDK* and the *LibOVR* library. In particular, the communication mechanism of the simulator, required to exchange data between the simulated robot and user applications, was based on the Robot Operating System (ROS) framework that runs under Linux systems.

However, employing the above-mentioned libraries to program the considered devices makes the overall system not easily interfaced for Linux-based frameworks, as a full support and compatibility is granted only for Windows-based systems. For this reason, we modified the overall simulation system, to remove the ROS-dependent parts, and exploited the remote V-REP APIs[6], to interface the virtual environment with an external application. In Figure A.7b, the overall scheme of the proposed simulator is presented: our application communicates with the Geomagic and Oculus devices through their corresponding libraries *OpenHaptics* and *LibOVR*, to read the state of the device and acquire specific information (e.g., tool position and velocity), or send specific commands (e.g., rendering a given force feedback on the haptic tools). The application communicates with each device on a separated thread, to keep the corresponding data acquisition rate unaffected. At the beginning of each simulation, the application enables a remote communication with the external V-REP simulator, through the use of remote V-REP API `simxStart` and automatically starts the simulator environment through the remote API `simxStartSimulation`. Then, while the simulation is running, for each iteration of the simulation loop, the application asks for the current joint configuration of the tele-operated PSMs end ECM, along with the current images acquired by the vision sensor objects, mounted at the end-effector of the ECM to simulate the endoscopic camera. This information is retrieved through the `simxGetJointPosition` and `simxGetVisionSensorCharImage` remote V-REP APIs, respectively.

In the reminder of this section, we further detail the connection of the employed devices with our application and the V-REP environment.

A.2.1 Connecting the Oculus Rift device

When the ROS-dependent parts of the simulator are removed, the connection between the Oculus Rift SDK and V-REP can be initialized. The Oculus Rift hardware kit considers an infrared-based positional tracking system called Constellation (see Fig. A.8a). The Constellation device provides accurate high-rate measurements of the frame $\mathcal{F}_r$ attached to the HMD, expressed with respect to reference frame $\mathcal{F}_t$

[4]https://www.3dsystems.com/haptics-devices/touch
[5]https://www.oculus.com/
[6]http://www.coppeliarobotics.com/helpFiles/en/remoteApiOverview.htm

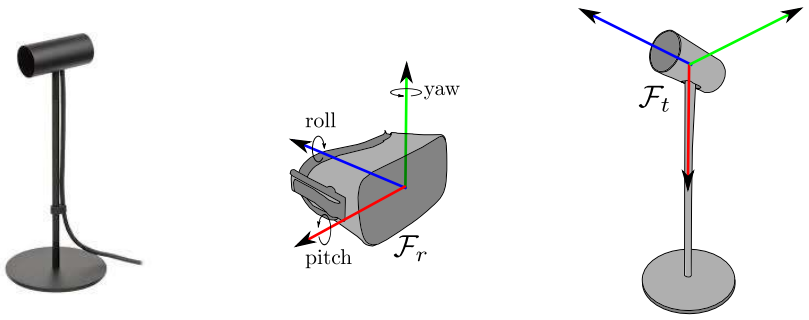


Figure A.8. (a) The Constellation device; (b) The reference frames $\mathcal{F}_r$ and $\mathcal{F}_t$ defined on the Oculus display and the Constellation tracking system, respectively.

attached to the Constellation device (see Fig. A.8b). To properly tele-operate the end-effector (i.e., the endoscopic cameras) of the ECM through the movements of the Oculus HMD, the corresponding velocities have to be kept consistent. In detail, we first query the *LibOVR* library to extract the 6D velocity vector ${}^t\boldsymbol{v}_r = \begin{bmatrix} {}^t\mathbf{v}_r^T & {}^t\boldsymbol{\omega}_r^T \end{bmatrix}^T$, denoting the linear and angular velocity of the Oculus display expressed in $\mathcal{F}_t$, along with the rotation matrix ${}^t\boldsymbol{R}_r$, expressing the orientation of the Oculus display with respect to $\mathcal{F}_t$. Once the data are available, we get the 6D Oculus display velocity vector expressed in its own frame $\mathcal{F}_r$:

$$ {}^r\boldsymbol{v}_r = \begin{bmatrix} {}^t\boldsymbol{R}_r^T & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^t\boldsymbol{R}_r^T \end{bmatrix} {}^t\boldsymbol{v}_r \tag{A.1} $$

To consistently generate the joint velocity vector $\dot{\boldsymbol{q}}_{\mathcal{C}} = [\dot{q}_1 \ \dot{q}_2 \ \dot{q}_3 \ \dot{q}_4]^T$ of the ECM, we need to consider the corresponding 6×4 Jacobian matrix $\boldsymbol{J}_{\mathcal{C}} = [\boldsymbol{J}_{\boldsymbol{v}}, \boldsymbol{J}_{\boldsymbol{\omega}}]^T$ in the end-effector frame $\mathcal{C}$, that can be easily reconstructed from the direct kinematics summarized in the DH Table shown in Table A.3. Finally, we consider a fixed rotation matrix ${}^{\mathcal{C}}\boldsymbol{R}_r$, to correctly match the motion directions of the Oculus Rift display with the endoscopic cameras of the ECM. Therefore, the 6D vector that maps the user's head motion to the end-effector of the ECM is given by

$$ {}^{\mathcal{C}}\boldsymbol{v}_{\mathcal{C}} = \begin{bmatrix} {}^{\mathcal{C}}\mathbf{v}_{\mathcal{C}} \\ {}^{\mathcal{C}}\boldsymbol{\omega}_{\mathcal{C}} \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{C}}\boldsymbol{R}_r & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{\mathcal{C}}\boldsymbol{R}_r \end{bmatrix} {}^r\boldsymbol{v}_r \tag{A.2} $$

However, since the ECM is a 4-DoF manipulator, we cannot assign an arbitrary 6D Cartesian velocity to the cameras, but only 4 out of the 6 space dimensions. For this reason, we implemented a user-enabled switching mechanism to alternatively control: (i) the orientation of the cameras, through the three revolute joints of the arm; (ii) the position along the longitudinal axis of the arm, corresponding to the z-axis of the camera frame $\mathcal{C}$, through the prismatic joint of the arm (as shown in Fig. A.3a). This assignment is equivalent to command the cameras in an *open-loop* control scheme, as no feedback on the current velocity of the end-effector is taken into account. This is a reasonable assumption, motivated by the presence of a human in the loop, able to correct possible actuation inaccuracies guaranteeing accurate

head motion tracking by cameras. Therefore, we designed a decoupled control of the position and the orientation of the ECM cameras as follows:

$$\begin{cases} \dot{\boldsymbol{q}}_{1,2,4} = \boldsymbol{J}_{\omega}^{\#}\ {}^{\mathcal{C}}\boldsymbol{\omega}_{\mathcal{C}} & \text{, if orientation control enabled} \\ \dot{q}_3 = {}^{\mathcal{C}}\mathrm{v}_{\mathcal{C},z} & \text{, if position control enabled} \end{cases} \tag{A.3}$$

being $\dot{\boldsymbol{q}}_{1,2,4} = [\dot{q}_1, \dot{q}_2, \dot{q}_4]^T$ the vector of the revolute joint velocities and $\dot{q}_3$ the prismatic joint velocity, while $\boldsymbol{J}_{\omega}^{\#}$ denotes the pseudo-inverse matrix of $\boldsymbol{J}_{\omega}$. We assumed that the user chooses which control has to be enabled, through a keyboard input. When choosing a given control, the joint velocities involved in the unused scheme are set to 0. The overall content implementing the interfacing of the application with the Oculus Rift is realized in our application on the separate thread, referred as `riftCallback`. Within the scope of this function, a proper OpenGL context is created to correctly render the images acquired from the pair of vision sensor objects in the V-REP scene. Then, the values of joint velocities are computed as described above, and used to directly command the joint position of the ECM in the V-REP environment, through the remote API `simxSetJointPosition`.

A.2.2 Connecting the Geomagic Touch devices

The communication between the pair of Geomagic Touch haptic devices and the left and right PSM of the da Vinci system is achieved in an analogous way. Each Geomagic Touch device is a 6DoF interface equipped with joint encoders that measure the full pose of the Haptic Interface Point (HIP) of the stylus held by the user (see Figure A.9. In addition, the devices provide a 3-DoF force feedback, which allows the user to touch and manipulate virtual objects, or reproduce physical contacts of tele-operated objects.

Similarly to the procedure described for the Oculus Rift and the ECM, the velocity vectors of the PSM end-effector (i.e, the grippers) and the HIP of the haptic device have to be kept consistent, to properly tele-operate the PSMs through the movements of the stylus. In detail, with reference to Figure A.9 and for each haptic device, we query the *OpenHaptics* library to extract the 6D velocity vector ${}^{\mathcal{BG}}\boldsymbol{v}_{\mathcal{H}}$ and the rotation matrix ${}^{\mathcal{BG}}\boldsymbol{R}_{\mathcal{H}}$, denoting the current linear and angular velocity and the orientation of the reference frame $\mathcal{F}_{\mathcal{H}}$, attached to the HIP of the device, in the base frame $\mathcal{F}_{\mathcal{BG}}$ of the Geomagic Touch. Therefore, we compute the 6D velocity vector of the HIP, expressed in its own frame $\mathcal{F}_{\mathcal{H}}$:

$${}^{\mathcal{H}}\boldsymbol{v}_{\mathcal{H}} = \begin{bmatrix} {}^{\mathcal{BG}}\boldsymbol{R}_{\mathcal{H}}^T & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & {}^{\mathcal{BG}}\boldsymbol{R}_{\mathcal{H}}^T \end{bmatrix} {}^{\mathcal{BG}}\boldsymbol{v}_{\mathcal{H}} \tag{A.4}$$

To determine the desired joint velocity $\dot{\boldsymbol{q}}_p$ of the considered PSM, we compute the corresponding 6×6 Jacobian matrix $\boldsymbol{J}_{\mathcal{G}}$ in the gripper frame $\mathcal{F}_{\mathcal{G}}$, reconstructed from the DH Tabls shown in Table A.2. Finally, to match the motion directions of the HIP, commanded by the user, with the motion of the gripper, we consider a rotation matrix ${}^{\mathcal{G}}\boldsymbol{R}_{\mathcal{H}}$ that takes into account the orientation of the gripper frame $\mathcal{F}_{\mathcal{G}}$ shown in Figure A.3b. Specifically, we compute the reference gripper velocity

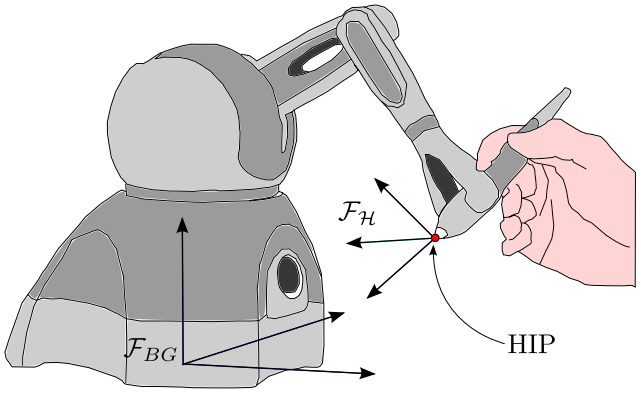


Figure A.9

Figure A.10. Reference frames of interest, where the Haptic Interface Point (HIP) is highlighted.

${}^{\mathcal{G}}\boldsymbol{v}_{\mathcal{G}}$, expressed in $\mathcal{F}_{\mathcal{G}}$ as

$$ {}^{\mathcal{G}}\boldsymbol{v}_{\mathcal{G}} = \begin{bmatrix} {}^{\mathcal{G}}\mathbf{v}_{\mathcal{G}} \\ {}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{G}} \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{G}}\boldsymbol{R}_{\mathcal{H}} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & {}^{\mathcal{G}}\boldsymbol{R}_{\mathcal{H}} \end{bmatrix} {}^{\mathcal{H}}\boldsymbol{v}_{\mathcal{H}} \tag{A.5} $$

In this case, the 7DoFs of the PSM (with the last one demanded to command the opening and closure of the gripper) allows to command a ful 6D desired Cartesian velocity. Therefore, the tele-operation of the PSM through the *open-loop* controls sent by the haptic device is achieved as

$$ \dot{\boldsymbol{q}}_p = \boldsymbol{J}_{\mathcal{G}}^{-1} \, {}^{\mathcal{G}}\boldsymbol{v}_{\mathcal{G}} \tag{A.6} $$

As for the control of the ECM through the Oculus Rift, we remark that the choice of the open-loop control adopted is motivated by the statement that the human user is present in the control scheme, and can directly compensate possible regulation errors of the grippers.

A typical issue in tele-operation tasks is the geometrical heterogeneity between master and slave workspaces (i.e., the haptic device and the PSM of the da Vinci system, respectively). Specifically, the Gemoagic Touch has a limited workspace, due to the short length of the links and the finite positional ranges of the joints. However, the size and the kinematic chain of the PSM is different, thus also the corresponding workspace in which the end-effector (i.e., the gripper of the PSM) moves is distinct. A common workaround that handles this discrepancy considers the use of a *clutch*-based mechanism, to enable/disable the tele-operation of the slave with the master device upon explicit command of the user. This way, when the HIP of the Geomagic Touch has reached the workspace limits of the device, the tele-operation can be disabled and the user can purposely relocate the stylus in a favourable configuration to further move the HIP in the desired direction. This behaviour is implemented through one of the buttons mounted on the stylus of the Geomagic Touch.

The related components of the proposed application, implementing the communication with the Geomagic Touch devices and the computation of the desired joint

velocity values, are realized in a separate thread, within the function `hapticCallback`. Within this function, both devices are queried to retrieve the current state of the HIP through the related *OpenHaptics* APIs. From the retrieved data, the desired joint velocity vector $\dot{\boldsymbol{q}}_p$ is generated and used to directly command the simulated PSM of the da Vinci system in the V-REP environment, through the remote API `simxSetJointPosition`.

A.2.3 Simulation results

To show the effectiveness of the simulator and the tele-operation scheme described in the previous sections, we considered the *pick-and-place* simulated training scenario, already introduced in Section A.1.3.

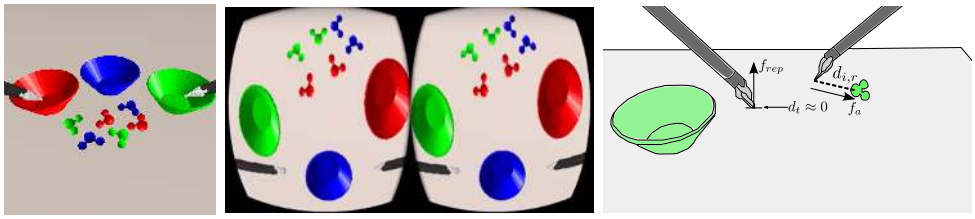
The scene has been realized by developing and importing CAD models of a training setup, composed of a set of small rigid objects and three cups, into the scene (see Figure A.11a). These objects are positioned on a table placed in front of the simulated da Vinci system.

The goal of the training is to move the pair of PSMs through the Geomagic Touch devices, in order to grasp and place the little coloured objects in the corresponding cup with the same colour. The entire scene is observed by the user through the Oculus Rift HMD, showing the images acquired by the ECM positioned on top of the table (see Figure A.11b). Contacts and interactions among objects have been simulated by creating respondable and simplified dynamic entities through the embedded V-REP functions.

To help the user in the satisfaction of this task and increase his sensory experience, we implemented additional features that exploit the functionalities of the simulator and the devices connected to the system, as depicted in Figure A.11c. Specifically, we exploited the distance calculation module of V-REP to compute, at each simulation time step, the distance $d_{i,s}$ ($s = \{l, r\}$ based on which PSM is considered) of the i-th grasping object from both the grippers. Among all the possible $d_{i,s}$, we evaluate the minimum distance and, if it goes below a given threshold, render an attractive force f_a directed towards the object. When the object is sufficiently close for grasping, we exploit the second button placed on the surface of the stylus of the Geomagic Touch device, to enable the grasping and attach the object to the gripper of the considered PSM. At this point, tele-operating the PSM allows to move the grasped object at any location and place it over the corresponding cup. Finally, the user can release the grasp and let the object fall in the cup, by pressing again the second button of the stylus.

On the other hand, for demonstration purposes, we also implemented a repulsive force f_{rep} on the planar surface of the table, to give the user the tactile experience of a contact of the gripper with a highly rigid object. Future developments of the simulator will consider the possibility to physically interact in a realistic way also with the other objects of the scene.

This simulator has been also presented to the Maker Faire 2018 of Rome. During the event, several people of different ages, mainly non-expert users, proposed to test the simulator and experience the training scenario, as depicted in Figure A.12. Despite some initial difficulties due to the lack of confidence with the system and the equipped devices, most of them were able to successfully tele-operate the PSM



(a) **(b)** **(c)**

Figure A.11. (a) Pick-and-place training scene. (b) Representation of the functionalities implemented in the training scenario.

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

(e) **(f)** **(g)** **(h)**

Figure A.12. Workstation and several non-expert users testing the *portable* da Vinci Simulator at Maker Faire 2018 of Rome.

and the ECM of the da Vinci system through the pair of Geomagic Touch devices and the Oculus Rift display, declaring their satisfaction for the impressive immersive experience.

Bibliography

[1] T. S. Collett, D. D. Lent, and P. Graham, "Scene perception and the visual control of travel direction in navigating wood ants," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 369, no. 1636, p. 20130035, 2014.

[2] J. Zeil, A. Narendra, and W. Stürzl, "Looking and homing: how displaced ants decide where to go," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 369, no. 1636, p. 20130034, 2014.

[3] M. V. Srinivasan, "How bees exploit optic flow: behavioural experiments and neural models," *Phil. Trans. R. Soc. Lond. B*, vol. 337, no. 1281, pp. 253–259, 1992.

[4] T. W. Cronin, M. J. Bok, N. J. Marshall, and R. L. Caldwell, "Filtering and polychromatic vision in mantis shrimps: themes in visible and ultraviolet vision," *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, vol. 369, no. 1636, p. 20130032, 2014.

[5] K. Steich, M. Kamel, P. Beardsley, M. K. Obrist, R. Siegwart, and T. Lachat, "Tree cavity inspection using aerial robots," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 4856–4862.

[6] S. M. Lavalle, "Rapidly-exploring random trees: A new tool for path planning," Tech. Rep., 1998.

[7] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 5, pp. 501–518, Oct 1992.

[8] F. Chaumette and S. Hutchinson, "Visual servo control. i. basic approaches," *IEEE Robotics & Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[9] A. Khelloufi, N. Achour, R. Passama, and A. Cherubini, "Tentacle-based moving obstacle avoidance for omnidirectional robots with visibility constraints," 2017, pp. 1331–1336.

[10] C. Sprunk, B. Lau, P. Pfaff, and W. Burgard, "An accurate and efficient navigation system for omnidirectional robots in industrial environments," *Autonomous Robots*, vol. 41, no. 2, pp. 473–493, 2017.

[11] A. Schaub, D. Baumgartner, and D. Burschka, "Reactive obstacle avoidance for highly maneuverable vehicles based on a two-stage optical flow clustering," *IEEE Trans. Intell. Transport. Syst.*, vol. 18, no. 8, pp. 2137–2152, 2017.

[12] O. Stasse, "SLAM and vision-based humanoid navigation," 2018.

[13] S. Oßwald, P. Karkowski, and M. Bennewitz, "Efficient coverage of 3d environments with humanoid robots using inverse reachability maps," in *2017 IEEE-RAS 17th IEEE-RAS International Conference on Humanoid Robots*, 2017, pp. 151–157.

[14] D. Maier, C. Stachniss, and M. Bennewitz, "Vision-based humanoid navigation using self-supervised obstacle detection," *Int. Journal of Humanoid Robotics*, vol. 10, no. 2, 2013.

[15] A. Paolillo, A. Faragasso, G. Oriolo, and M. Vendittelli, "Vision-based maze navigation for humanoid robots," *Autonomous Robots*, vol. 41, no. 2, pp. 293–309, 2017.

[16] J. Delfin, H. M. Becerra, and G. Arechavaleta, "Humanoid navigation using a visual memory with obstacle avoidance," *Robotics and Autonomous Systems*, vol. 109, pp. 109–124, 2018.

[17] M. Ferro, A. Paolillo, A. Cherubini, and M. Vendittelli, "Omnidirectional humanoid navigation in cluttered environments based on optical flow information," in *IEEE-RAS International Conference on Humanoid Robots*, 2016, pp. 75–80.

[18] ——, "Vision-based navigation for omnidirectional mobile robots," 2018.

[19] A. Paolillo, A. Faragasso, G. Oriolo, and M. Vendittelli, "Vision-based maze navigation for humanoid robots," *Autonomous Robots*, 2016.

[20] N. Ohnishi and A. Imiya, "Appearance-based navigation and homing for autonomous mobile robot," *Image and Vision Computing*, vol. 31, no. 6, pp. 511–532, 2013.

[21] B. D. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[22] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Scandinavian conference on Image analysis*. Springer, 2003, pp. 363–370.

[23] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision*. Springer, 2004, pp. 25–36.

[24] F. Chaumette, "Image moments: a general and useful set of features for visual servoing," *IEEE Transactions on Robotics*, vol. 20, no. 4, pp. 713–723, Aug 2004.

[25] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*. Springer Verlag, 2003.

[26] G. Oriolo, A. Paolillo, L. Rosa, and M. Vendittelli, "Humanoid odometric localization integrating kinematic, inertial and visual information," *Autonomous Robots*, vol. 40, no. 5, Jun 2016. [Online]. Available: https://doi.org/10.1007/s10514-015-9498-0

[27] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[28] R. Barbosa, J. Silva, M. M. Junior, and R. Gallis, "Velocity estimation of a mobile mapping vehicle using filtered monocular optical flow," in *Int. Symp. on Mobile Mapping Technology*, vol. 5, 2007.

[29] A. Paolillo, P. Gergondet, A. Cherubini, M. Vendittelli, and A. Kheddar, "Autonomous car driving by a humanoid robot," *Journal of Field Robotics*, vol. 35, no. 2, pp. 169–186, 2018.

[30] B. Navarro, A. Cherubini, A. Fonte, G. Poisson, and P. Fraisse, "A framework for intuitive collaboration with a mobile manipulator," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 6293–6298.

[31] C. Dune, A. Herdt, O. Stasse, P.-B. Wieber, K. Yokoi, and E. Yoshida, "Cancelling the sway motion of dynamic walking in visual servoing," 2010, pp. 3175–3180.

[32] G. Oriolo, A. Paolillo, L. Rosa, and M. Vendittelli, "Vision-based trajectory control for humanoid navigation," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2013, pp. 118–123.

[33] H. Delingette, X. Pennec, L. Soler, J. Marescaux, and N. Ayache, "Computational models for image-guided robot-assisted and simulated medical interventions," *Proceedings of the IEEE*, vol. 94, no. 9, pp. 1678–1688, Sept 2006.

[34] H. Barbeau, M. Wainberg, and L. Finch, "Description and application of a system for locomotor rehabilitation," *Medical and Biological Engineering and Computing*, vol. 25, no. 3, pp. 341–344, 1987.

[35] P. Dario, M. Bergamasco, D. Femi, A. Fiorillo, and A. Vaccarelli, "Tactile perception in unstructured environments: A case study for rehabilitative robotics applications," in *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4, March 1987, pp. 2047–2054.

[36] E. Colgate and N. Hogan, "An analysis of contact instability in terms of passive physical equivalents," in *Proceedings, 1989 International Conference on Robotics and Automation*, May 1989, pp. 404–409 vol.1.

[37] C. Cozean, W. S. Pease, and S. Hubbell, "Biofeedback and functional electric stimulation in stroke rehabilitation." *Archives of physical medicine and rehabilitation*, vol. 69, no. 6, pp. 401–405, 1988.

[38] H. Kwee, J. Duimel, J. Smit, A. De Moed, J. Van Woerden, and L. Kolk, "The manus wheelchair-mounted manipulator: developments toward a production model," in *Proc. 3rd Int. Conf. Assoc. Advancement Rehab. Technol*, 1988, pp. 460–462.

[39] H. I. Krebs, N. Hogan, M. L. Aisen, and B. T. Volpe, "Robot-aided neurorehabilitation," *IEEE Transactions on Rehabilitation Engineering*, vol. 6, no. 1, pp. 75–87, March 1998.

[40] J. F. Veneman, R. Kruidhof, E. E. Hekman, R. Ekkelenkamp, E. H. Van Asseldonk, and H. Van Der Kooij, "Design and evaluation of the lopes exoskeleton robot for interactive gait rehabilitation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 15, no. 3, pp. 379–386, 2007.

[41] M. H. Rahman, M. J. Rahman, O. Cristobal, M. Saad, J.-P. Kenné, and P. S. Archambault, "Development of a whole arm wearable robotic exoskeleton for rehabilitation and to assist upper limb movements," *Robotica*, vol. 33, no. 1, pp. 19–39, 2015.

[42] P. Polygerinos, Z. Wang, K. C. Galloway, R. J. Wood, and C. J. Walsh, "Soft robotic glove for combined assistance and at-home rehabilitation," *Robotics and Autonomous Systems*, vol. 73, pp. 135–143, 2015.

[43] R. Moesges and S. Lavallee, "Computer integrated surgery," *Computer-integrated surgery: technology and clinical applications*, p. 5, 1996.

[44] T. Kanade, B. Davies, and C. N. Riviere, "Special issue on medical robotics," *Proceedings of the IEEE*, vol. 94, no. 9, pp. 1649–1651, 2006.

[45] J. Troccaz, M. Baumann, P. Berkelman, P. Cinquin, V. Daanen, A. Leroy, M. Marchal, Y. Payan, E. Promayon, S. Voros, *et al.*, "Medical image computing and computer-aided medical interventions applied to soft tissues: Work in progress in urology," *Proceedings of the IEEE*, vol. 94, no. 9, pp. 1665–1677, 2006.

[46] N. Sugano, "Computer-assisted orthopedic surgery," *Journal of Orthopaedic Science*, vol. 8, no. 3, pp. 442–448, 2003.

[47] R. Klette, *Concise computer vision*. Springer, 2014.

[48] A. Ayvaci, P. Yan, S. Xu, S. Soatto, and J. Kruecker, "Biopsy needle detection in transrectal ultrasound," *Computerized Medical Imaging and Graphics*, vol. 35, no. 7, pp. 653–659, 2011.

[49] G. J. Vrooijink, M. Abayazid, and S. Misra, "Real-time three-dimensional flexible needle tracking using two-dimensional ultrasound," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 1688–1693.

[50] M. Waine, C. Rossa, R. Sloboda, N. Usmani, and M. Tavakoli, "Needle tracking and deflection prediction for robot-assisted needle insertion using 2d ultrasound images," *Journal of Medical Robotics Research*, vol. 1, no. 01, p. 1640001, 2016.

[51] J. Hong, T. Dohi, M. Hashizume, K. Konishi, and N. Hata, "An ultrasound-driven needle-insertion robot for percutaneous cholecystostomy," *Physics in Medicine and Biology*, vol. 49, no. 3, p. 441, 2004.

[52] K. Mathiassen, D. Dall'Alba, R. Muradore, P. Fiorini, and O. J. Elle, "Real-time biopsy needle tip estimation in 2d ultrasound images," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. IEEE, 2013, pp. 4363–4369.

[53] S. H. Okazawa, R. Ebrahimi, J. Chuang, R. N. Rohling, and S. E. Salcudean, "Methods for segmenting curved needles in ultrasound images," *Medical image analysis*, vol. 10, no. 3, pp. 330–342, 2006.

[54] A. Casals, J. Amat, D. Prats, and E. Laporte, "Vision guided robotic system for laparoscopic surgery," in *Proc. of the IFAC International Congress on Advanced Robotics*, 1995, pp. 33–36.

[55] G.-Q. Wei, K. Arbter, and G. Hirzinger, "Automatic tracking of laparoscopic instruments by color coding," in *CVRMed-MRCAS'97*. Springer, 1997, pp. 357–366.

[56] A. Krupa, M. de Mathelin, C. Doignon, J. Gangloff, G. Morel, L. Soler, J. Leroy, and J. Marescaux, "Automatic 3-d positioning of surgical instruments during robotized laparoscopic surgery using automatic visual feedback," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2002, pp. 9–16.

[57] C. Doignon, P. Graebling, and M. De Mathelin, "Real-time segmentation of surgical instruments inside the abdominal cavity using a joint hue saturation color feature," *Real-Time Imaging*, vol. 11, no. 5, pp. 429–442, 2005.

[58] S. Speidel, A. Kroehnert, S. Bodenstedt, H. Kenngott, B. Mueller-Stich, and R. Dillmann, "Image-based tracking of the suturing needle during laparoscopic interventions," in *SPIE Medical Imaging*. International Society for Optics and Photonics, 2015, pp. 94 150B–94 150B.

[59] C. Wengert, L. Bossard, A. Häberling, C. Baur, G. Székely, and P. C. Cattin, "Endoscopic navigation for minimally invasive suturing," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2007, pp. 620–627.

[60] P. Hynes, G. Dodds, and A. Wilkinson, "Uncalibrated visual-servoing of a dual-arm robot for mis suturing," in *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.* IEEE, 2006, pp. 420–425.

[61] C. Staub, T. Osa, A. Knoll, and R. Bauernschmitt, "Automation of tissue piercing using circular needles and vision guidance for computer aided laparoscopic surgery," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 4585–4590.

[62] J. Chang, "Robust needle recognition using Artificial Neural Network (ANN) and Random Sample Consensus (RANSAC)," *IEEE Applied Imagery Pattern Recognition Workshop*, pp. 1–3, 2012.

[63] F. Zhong, D. Navarro-alarcon, Z. Wang, Y.-h. Liu, T. Zhang, H. M. Yip, and H. Wang, "Adaptive 3D Pose Computation of Suturing Needle Using Constraints From Static Monocular Image Feedback," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016, pp. 5521–5526.

[64] C. Wengert, L. Bossard, C. Baur, G. Székely, and P. C. Cattin, "Endoscopic navigation for minimally invasive suturing." *Journal of the International Society for Computer Aided Surgery*, vol. 13, no. 5, pp. 299–310, 2008. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/18821347

[65] S. Speidel, A. Kroehnert, S. Bodenstedt, H. Kenngott, B. Müller-Stich, and R. Dillmann, "Image-based tracking of the suturing needle during laparoscopic interventions," *Medical Imaging: Image-Guided Procedures, Robotic Interventions, and Modeling*, vol. 9415, p. 94150B, 2015.

[66] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[67] D. López de Ipiña, P. R. Mendonça, and A. Hopper, "Trip: A low-cost vision-based location system for ubiquitous computing," *Personal and Ubiquitous Computing*, vol. 6, no. 3, pp. 206–219, 2002.

[68] N. Trawny and S. I. Roumeliotis, "Indirect kalman filter for 3d attitude estimation," *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep*, vol. 2, p. 2005, 2005.

[69] G. A. Fontanelli, M. Selvaggio, M. Ferro, F. Ficuciello, M. Vendiuelli, and B. Siciliano, "A v-rep simulator for the da vinci research kit robotic platform," in *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob)*, Aug 2018, pp. 1056–1061.

[70] A. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least square fitting of ellipses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 476–480, 1999.

[71] "da Vinci research kit (DVRK) wiki." [Online]. Available: http://research.intusurg.com/dvrkwiki

[72] P. Salaris, R. Spica, P. R. Giordano, and P. Rives, "Online optimal active sensing control," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 672–678.

[73] M. Cognetti, P. Salaris, and P. R. Giordano, "Optimal active sensing with process and measurement noise," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 2118–2125.

[74] G. Costante, J. Delmerico, M. Werlberger, P. Valigi, and D. Scaramuzza, "Exploiting photometric information for planning under uncertainty," in *Robotics Research*. Springer, 2018, pp. 107–124.

[75] R. Spica and P. R. Giordano, "A framework for active estimation: Application to structure from motion," in *52nd IEEE Conference on Decision and Control*, Dec 2013, pp. 7647–7653.

[76] A. Mateus, O. Tahri, and P. Miraldo, "Active structure-from-motion for 3d straight lines," *arXiv preprint arXiv:1807.00753*, 2018.

[77] R. Spica, P. R. Giordano, and F. Chaumette, "Active structure from motion: Application to point, sphere, and cylinder," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1499–1513, Dec 2014.

[78] ——, "Active structure from motion for spherical and cylindrical targets," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 5434–5440.

[79] P. R. Giordano, R. Spica, and F. Chaumette, "An active strategy for plane detection and estimation with a monocular camera," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 4755–4761.

[80] R. Spica, P. R. Giordano, and F. Chaumette, "Plane estimation by active vision from point features and image moments," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 6003–6010.

[81] O. Tahri, P. R. Giordano, and Y. Mezouar, "Rotation free active vision," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 3086–3091.

[82] R. Spica, P. R. Giordano, and F. Chaumette, "Coupling visual servoing with active structure from motion," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 3090–3095.

[83] ——, "Coupling active depth estimation and visual servoing via a large projection operator," *The International Journal of Robotics Research*, vol. 36, no. 11, pp. 1177–1194, 2017. [Online]. Available: https://doi.org/10.1177/0278364917728327

[84] D. J. Agravante and F. Chaumette, "Active vision for pose estimation applied to singularity avoidance in visual servoing," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 2947–2952.

[85] C. Collewet, E. Marchand, and F. Chaumette, "Visual servoing set free from image processing," in *2008 IEEE International Conference on Robotics and Automation*, May 2008, pp. 81–86.

[86] C. Collewet and E. Marchand, "Photometric visual servoing," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 828–834, Aug 2011.

[87] L.-A. Duflot, R. Reisenhofer, B. Tamadazte, N. Andreff, and A. Krupa, "Wavelet and shearlet-based image representations for visual servoing," *The International Journal of Robotics Research*, p. 0278364918769739, 2018.

[88] N. Crombez, G. Caron, and E. M. Mouaddib, "Photometric gaussian mixtures based visual servoing," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 5486–5491.

[89] C. Teulière and E. Marchand, "A dense and direct approach to visual servoing using depth maps," *IEEE Transactions on Robotics*, vol. 30, no. 5, pp. 1242–1249, Oct 2014.

[90] M. Bakthavatchalam, O. Tahri, and F. Chaumette, "A direct dense visual servoing approach using photometric moments," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1226–1239, Oct 2018.

[91] P. R. Giordano, R. Spica, and F. Chaumette, "Learning the shape of image moments for optimal 3d structure estimation," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 5990–5996.

[92] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da vinci surgical system," in *2014 IEEE Int. Conf. on Robotics and Automation*, May 2014, pp. 6434–6439.

[93] M. Selvaggio, G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano, "Passive virtual fixtures adaptation in minimally invasive robotic surgery," *IEEE Robotics and Automation Letters*, 2018.

[94] M. Selvaggio, G. Notomista, F. Chen, B. Gao, F. Trapani, and D. Caldwell, "Enhancing bilateral teleoperation using camera-based online virtual fixtures generation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016, pp. 1483–1488.

[95] J. M. Prendergast and M. E. Rentschler, "Towards autonomous motion control in minimally invasive robotic surgery," *Expert Review of Medical Devices*, vol. 13, no. 8, pp. 741–748, 2016.

[96] A. Moglia, V. Ferrari, L. Morelli, M. Ferrari, F. Mosca, and A. Cuschieri, "A Systematic Review of Virtual Reality Simulators for Robot-assisted Surgery," *Eur Urol*, pp. 1065–1080, 2016.

[97] A. Baheti, S. Seshadri, A. Kumar, G. Srimathveeravalli, T. Kesavadas, and K. Guru, "Ross: Virtual reality robotic surgical simulator for the da vinci surgical system," in *Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2008, pp. 479–480.

[98] "SimSurgery Educational Platform (SEP)." [Online]. Available: http://www.simsurgery.com

[99] "Mimic Simulation dV-Trainer." [Online]. Available: http://www.mimicsimulation.com/products/dv-trainer/

[100] "da Vinci Skills Simulator." [Online]. Available: https://www.intuitivesurgical.com/products/skills_simulator/

[101] "RobotiX Mentor Simbionix." [Online]. Available: simbionix.com/simulators/robotix-mentor/

[102] J. Sanchez-Margallo, J. P. Carrasco, L. Sanchez-Peralta, J. M. Cuevas, L. Gasperotti, D. Zerbato, and F. S.-M. L. Vezzaro, "A preliminary validation of the xron surgical simulator for robotic surgery," in *Int, Conf. of the Society for Medical Innovation and Technology*, 2013.

[103] R. Smith, M. Truong, and M. Perez, "Comparative analysis of the functionality of simulators of the da vinci surgical robot," *Surgical Endoscopy*, vol. 29, no. 4, pp. 972–983, Apr 2015.

[104] G. A. Fontanelli, M. Selvaggio, L. R. Buonocore, F. Ficuciello, L. Villani, and B. Siciliano, "A new laparoscopic tool with in-hand rolling capabilities for needle reorientation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2354–2361, 2018.

[105] G. A. Fontanelli, L. R. Buonocore, F. Ficuciello, L. Villani, and B. Siciliano, "A novel force sensing integrated into the trocar for minimally invasive robotic surgery," in *2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 131–136.

[106] "Smashing Robotics." [Online]. Available: https://goo.gl/gH8VE4

[107] "V-REP simulator." [Online]. Available: http://www.coppeliarobotics.com/

[108] G. Guthart and J. Salisbury, "The intuitive^{TM} telesurgery system: overview and application," in *IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 618–621.

[109] G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano, "Modelling and identification of the da Vinci research kit robotic arms," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 1464–1469.

[110] "John Hopkins University DVRK controller git repositories." [Online]. Available: https://github.com/jhu-dvrk

[111] Z. Chen, A. Deguet, R. H. Taylor, and P. Kazanzides, "Software architecture of the da vinci research kit," in *IEEE Int. Conf. on Robotic Computing*, 2017, pp. 180–187.