CENTRALE NANTES

THÈSE DE DOCTORAT DE

L'ÉCOLE CENTRALE DE NANTES

ÉCOLE DOCTORALE N° 602
Sciences pour l'Ingénieur
Spécialité : *Robotique - Mécanique*

Par
Zane ZAKE

Design and Stability Analysis of Visual Servoing on Cable-Driven Parallel Robots for Accuracy Improvement

Thése présentée et soutenue à Nantes, le 12/02/2021
Unité de recherche : UMR 6004, Laboratoire des Sciences du Numérique de Nantes (LS2N)

Rapporteurs avant soutenance :

Jean-Pierre MERLET	Directeur de recherche, INRIA Sophia-Antipolis
Marc GOUTTEFARDE	Directeur de recherche, CNRS, Montpellier

Composition du Jury :

Président :	Nicolas ANDREFF	Professeur des universités, Université de Franche-Comté, Besançon
Examinateurs :	Claire DUMAS	Ingénieure de recherche - Docteure, Cutii, Roubaix
	Nicolò PEDEMONTE	Ingénieur de recherche - Docteur, IRT Jules Verne, Bouguenais
Dir. de thèse :	Stéphane CARO	Directeur de recherche CNRS, École Centrale de Nantes
Co-dir. :	François CHAUMETTE	Directeur de recherche INRIA, Rennes

Acknowledgments

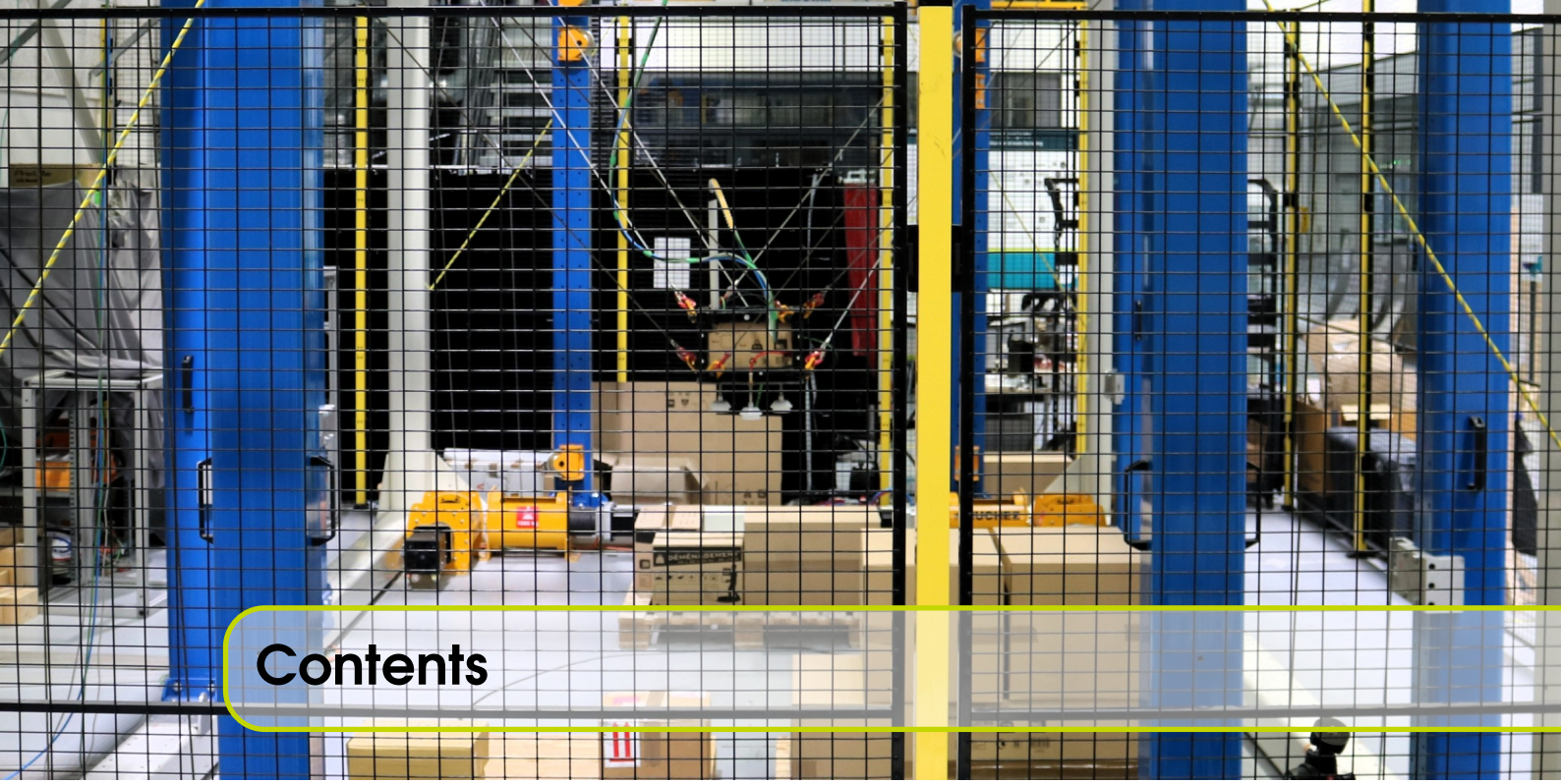
First, I would like to express my gratitude towards my supervisors: Stéphane Caro, François Chaumette and Nicolò Pedemonte. One might say that I got the supervisor jackpot by having supervisors with such vast experience, very different approaches to problem solving, passion for their work, endless amount of patience and giving me just the right amount of supervision. We set ambitious goals, and it was a pleasure working together to achieve them. And an additional thanks to Stéphane for proposing this thesis topic to me, as without him I would not have even known about this opportunity.

The thesis would not be possible without IRT Jules Verne and their Perform program. Indeed, this PhD thesis was proposed and subsequently financed as one of the very first within the Perform program. Furthermore, thanks to it I spent my time working side by side with the wonderful people in the Robotics and Cobotics team of IRT Jules Verne. I am grateful to everyone who took an interest in my thesis and offered help, advice or encouragement.

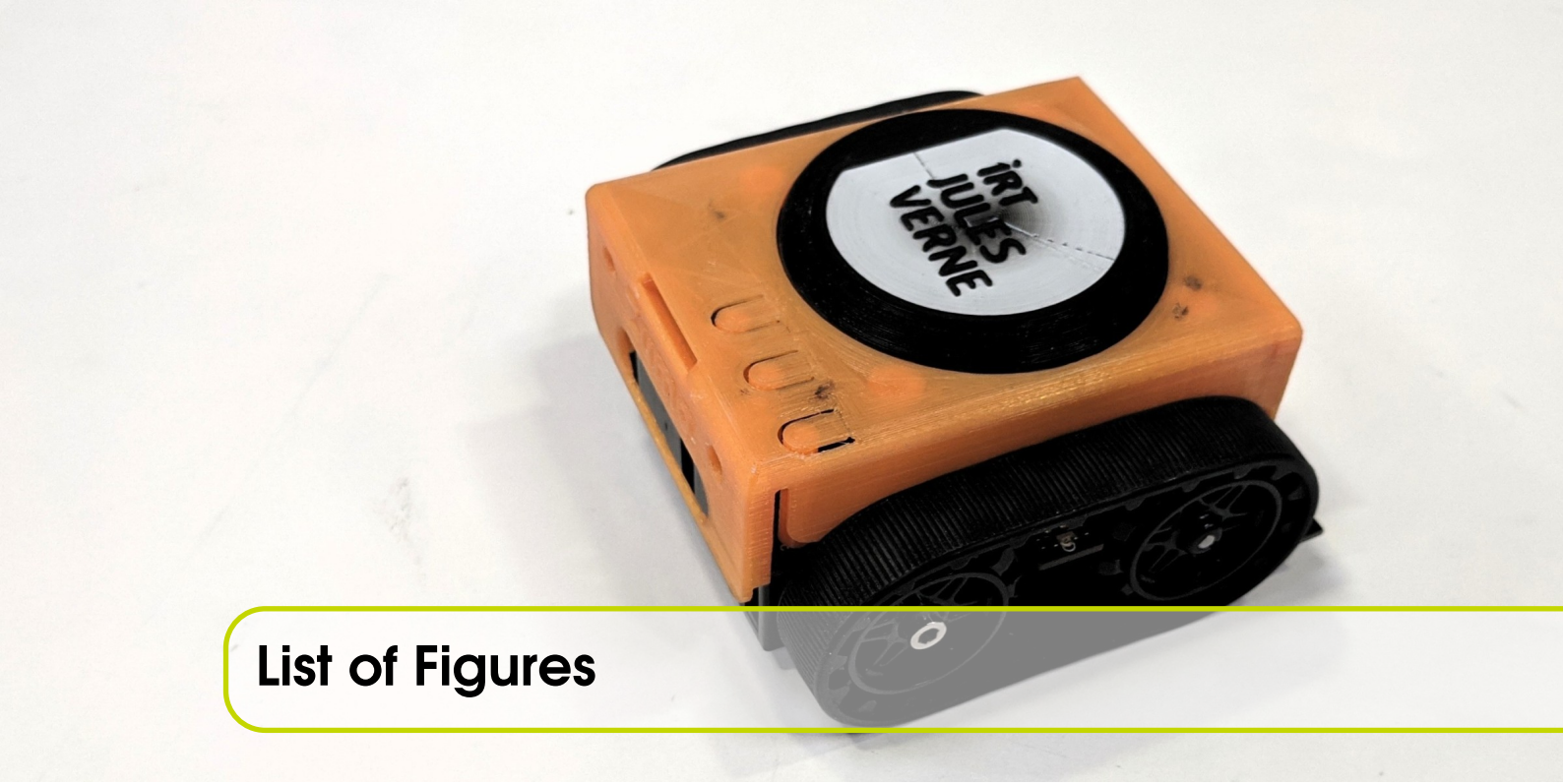
I am also grateful to the reviewers and the jury: Jean-Pierre Merlet, Marc Gouttefarde, Nicolas Andreff, and Claire Dumas. They evaluated the thesis and judged it worthy. Thank you for your questions and suggestions. And an especially big thank you to Jean-Pierre and Marc for writing such meticulous reviews of the manuscript and thus allowing me to improve it.

Finally, I want to thank my family. Thanks to my uncle Aleksejs Rovdo for always believing in me and supporting me. Thanks to my sisters Linda Zaķe, Veronika Taylor, Dana Markevica for rooting for me and organizing our video calls, allowing us to feel closer despite living in different countries. And most importantly, thank you to my partner

Mārtiņš Duboviks for moving with me to France so that I could pursue this degree, for patiently adapting our plans to my crazy work hours and ever pressing deadlines, and for surviving the lock-down being stuck within four walls with me and this manuscript. Thank you!

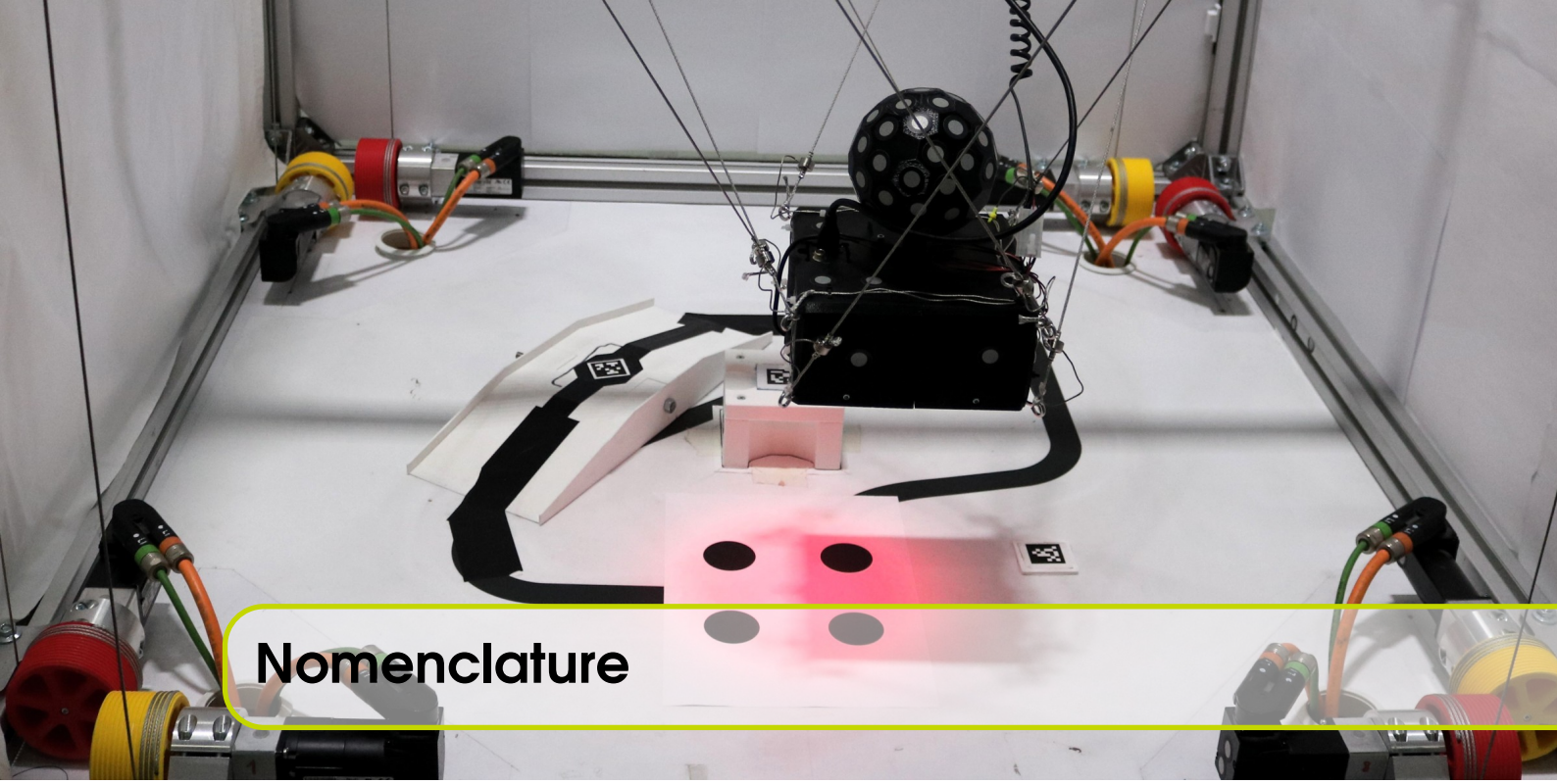


Contents



List of Figures

List of Tables



Nomenclature

Abbreviations

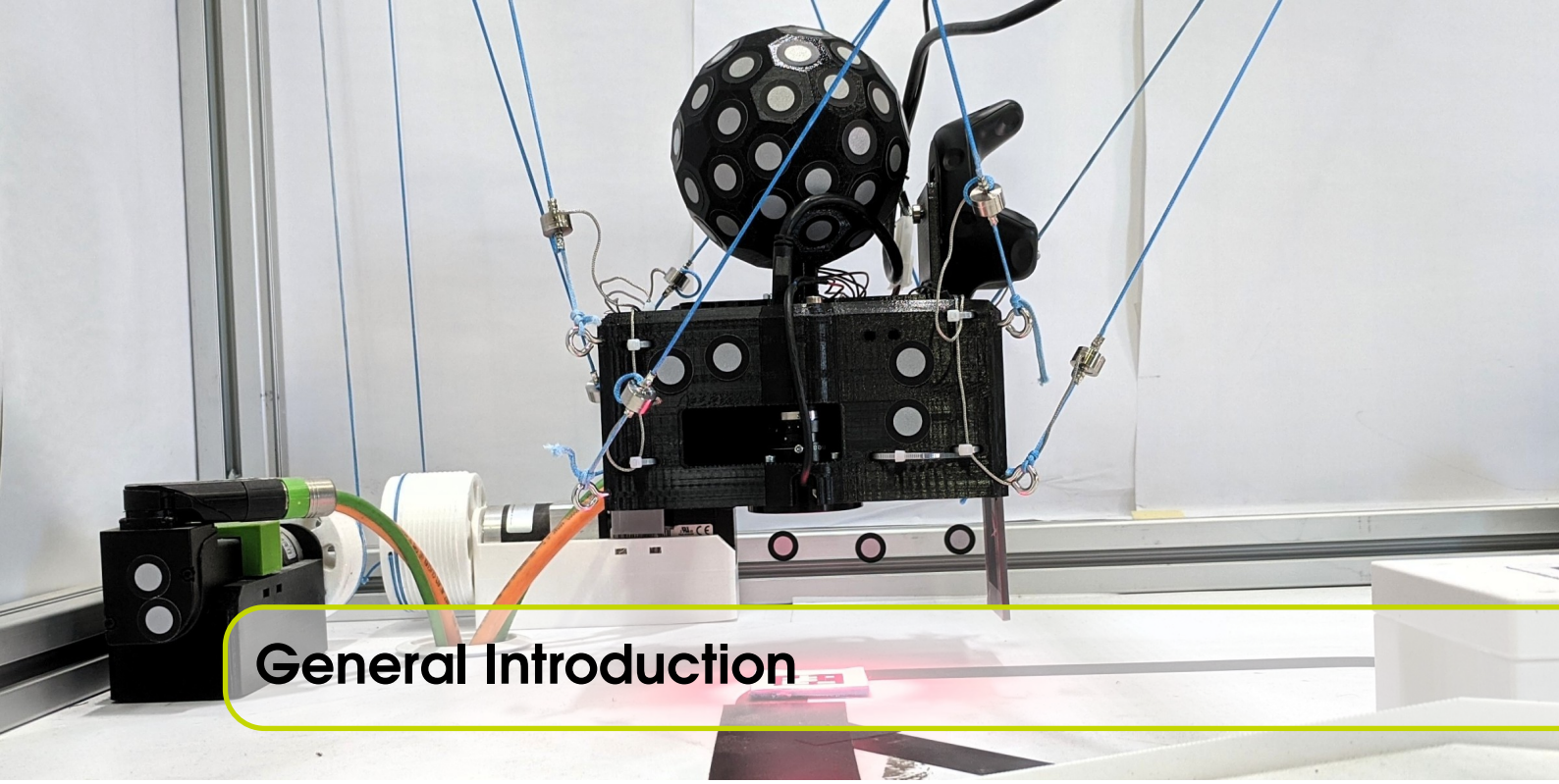
CDPR	–	Cable-Driven Parallel Robot
DoF	–	Degrees of Freedom
DGM	–	Direct Geometric Model
DGSM	–	Direct Geometrico-Static Model
IGM	–	Inverse Geometric Model
SFW	–	Static Feasible Workspace, also denoted as $\mathcal{S}$
TDA	–	Tension Distribution Algorithm
VS	–	Visual Servoing
IBVS	–	Image-Based Visual Servoing
PBVS	–	Pose-Based Visual Servoing
2½D VS	–	2½D Visual Servoing
GAS	–	Global Asymptotic Stability
LAS	–	Local Asymptotic Stability
CSW	–	Control Stability Workspace, also denoted as $\mathcal{Z}$
TCA	–	Tension Correction Algorithm

General Notation

Writing convention: (a) boldface lowercase characters denote vectors; (b) boldface uppercase characters denote matrices; (c) non-bold lowercase characters denote scalars; (d) non-bold uppercase characters denote points.

$\mathbf{B}^\top$	–	the transpose of $\mathbf{B}$
$\mathbf{B}^{-1}$	–	the inverse of $\mathbf{B}$
$\mathbf{B}^\dagger$	–	the pseudo-inverse of $\mathbf{B}$
$\widehat{\mathbf{B}}$	–	the estimation of $\mathbf{B}$
$[\mathbf{b}]_\times$	–	the cross-product matrix of vector $\mathbf{b}$
${}^i\mathbf{b}$	–	vector $\mathbf{b}$ expressed in $\mathcal{F}_i$
$\mathbf{b}^*$	–	the desired value of $\mathbf{b}$
$\mathbf{I}$	–	Identity matrix
n	–	number of the degrees of freedom of the moving-platform
m	–	number of cables
r	–	degree of redundancy
t	–	time
$\mathcal{C}_i$	–	the ith cable
$\mathcal{F}_i$	–	ith frame, for example $\mathcal{F}_b$ is the base frame
O_i	–	the origin of frame $\mathcal{F}_i$
${}^i\mathbf{t}_j$	–	(3×1) translational vector from $\mathcal{F}_i$ to $\mathcal{F}_j$
${}^i\mathbf{R}_j$	–	(3×3) rotation matrix from $\mathcal{F}_i$ to $\mathcal{F}_j$
${}^i\mathbf{T}_j$	–	(4×4) homogeneous transformation matrix from $\mathcal{F}_i$ to $\mathcal{F}_j$
$\mathbf{u}$	–	3-dimensional unit vector
$\theta\mathbf{u}$	–	axis-angle representation of rotation
${}^i\mathbf{p}_j$	–	pose of $\mathcal{F}_j$ expressed in $\mathcal{F}_i$, ${}^i\mathbf{p}_j = [{}^i\mathbf{t}_j \;\; \theta\mathbf{u}]$
${}^i\mathbf{v}_j$	–	velocity of $\mathcal{F}_j$ expressed in $\mathcal{F}_i$, ${}^i\mathbf{v}_j = [{}^i\boldsymbol{\nu}_j \;\; {}^i\boldsymbol{\omega}_\nu]$
$\mathbf{q}_m$	–	m-dimensional motor position vector
$\dot{\mathbf{q}}_m$	–	m-dimensional motor velocity vector
$\mathbf{l}$	–	m-dimensional cable length vector
$\dot{\mathbf{l}}$	–	m-dimensional cable velocity vector
A_i	–	ith cable exit point
B_i	–	ith cable anchor point

r_p	–	pulley radius
r_w	–	winch radius
$\mathbf{A}$	–	$(m \times n)$ Jacobian matrix that relates cable velocities $\dot{\mathbf{l}}$ to moving-platform velocity $\mathbf{v}_p$
$\mathbf{A}_d$	–	(6×6) Adjoint matrix used to express a velocity ${}^i\mathbf{v}_j$ in a different frame
$\boldsymbol{W}$	–	the wrench matrix, $\boldsymbol{W} = -\mathbf{A}^\top$
$\boldsymbol{w}_g$	–	6-dimensional gravity wrench vector
$\mathbf{t}_m$	–	m-dimensional motor torque vector
δl_i	–	elongation of the ith cable
E	–	Young modulus
$\boldsymbol{\tau}$	–	m-dimensional cable tension vector
$\delta \dot{l}_i$	–	velocity correction of ith cable from TCA
k_c	–	gain for $\delta \dot{l}_i$ computation
$\mathbf{s}$	–	j-dimensional feature vector with amount of features ideally $j = n$
$\boldsymbol{e}$	–	error between the current features $\mathbf{s}$ and the desired $\mathbf{s}^*$
λ	–	positive gain in visual servoing
$\mathbf{L_s}$	–	$(j \times 6)$ interaction matrix that relates camera velocity ${}^c\mathbf{v}_c$ and $\dot{\boldsymbol{e}}$
$\mathbf{o}$	–	object center-point coordinates in the image, $\mathbf{o} = [x_o, y_o]$
$\boldsymbol{\Pi}$	–	Stability criterion
$\mathcal{D}$	–	perturbation boundary
$\mathbf{d}$	–	perturbation set within $\mathcal{D}$



General Introduction

Robots have become an integral part of the industrial world. Automation of dull or dangerous tasks allows the factories to improve the work conditions for their employees and at the same time to increase the overall effectiveness of the plant. Furthermore, with the recent outbreak of Covid-19, the need for autonomous task execution that can be observed and interacted with off-site (or at least from a safe distance) only grows. New use cases, such as disinfection of factories, technical halls and even public transportation without putting the employee at risk, have emerged [Ver]. Different mechanical architectures of robots are being studied to better respond to the increasing and diversified demand for automation and robotization.

One promising type of robots with high potential is the Cable-Driven Parallel Robot (CDPR). As the name suggests it is a kind of parallel robots that has cables instead of rigid links. The moving-platform is connected to a base frame through the cables. Moving-platform motion is achieved by changing the cable lengths by winding and unwinding each cable on a winch, which is actuated by a rigidly fixed motor. CDPRs are characterized by a large workspace in translation, a large payload capacity and a low mass in motion. These advantages lead to interesting CDPR applications, such as: (a) moving large objects over large distances [ABD92] [Gag+16]; (b) moving objects with high velocity [Kaw+97]; (c) providing feedback for a virtual reality application [FCCG14]; (d) assisting human rescue operations [MD10]. However, currently available CDPRs are often lacking accuracy. In order to improve the accuracy of CDPRs one can enrich the mathematical models describing the geometric, kinematic, elasto-static and elasto-dynamic behavior of such robots while using ad hoc techniques for the calibration of these models [FCCG14]. In this case, the CDPR command is managed in its joint space. It

is also possible to use exteroceptive sensors such as precise cameras to control the CDPRs in the Cartesian space [RCM14] [Dal+19] [CCL15] [Beg+18].

This thesis focuses on the second approach since little work exists on the subject for the moment whereas it is in our opinion a relevant and promising approach to considerably improve the accuracy of CDPRs. Several configurations and their combinations can be used: sensors onboard the moving-platform [RCM14]; remote sensors observing the moving-platform [Dal+19] [CCL15] [Beg+18]; sensors observing the internals parts of the robots, such as cables [Dal+19]. To improve the accuracy of a CDPR with respect to an object, the camera is usually mounted on the moving-platform. In this configuration the camera and the moving-platform approach the object simultaneously. Moreover, as the camera approaches the object the accuracy is increased and at the desired state the accuracy is indeed excellent. However, with the camera onboard the moving-platform, its pose cannot be observed and thus must be estimated.

The kinematic models of the system comprising a CDPR and a camera have been determined and different visual servoing approaches have been studied. More precisely, we started with Pose-Based Visual Servoing (PBVS) [CH08]. Then, to improve the behavior of the robot, 2½D Visual Servoing (2½D VS) [MCB99] [KKC04] was implemented. The latter uses a combination of 3D information, such as the Cartesian pose of the object, and 2D information, such as object center-point coordinates in the image. Finally, Image-Based Visual Servoing (IBVS) [CH08] was also implemented.

As mentioned before, having the camera on the moving-platform, it is impossible to directly measure the moving-platform pose. Instead the pose has to be estimated. To find the best suited estimation method, three different approaches were studied: (a) control-based; (b) image-based; (c) model-based. In the control-based approach the control output is integrated in order to estimate the new moving-platform pose. In the image-based approach two images are used to measure the object pose in the camera frame in two time instants. It assumed that on one of these time instants the moving-platform pose is known. Then, assuming that the object does not move, the changes in the image are due to camera motion. It is thus possible to estimate the new moving-platform pose through frame transformation. Finally, in the model-based approach the lengths of the six cables that are most in tension are used to compute the moving-platform pose in the vicinity of the previous pose. In all of these approaches the initial moving-platform pose needs to be known and the more precise the knowledge, the better the overall behavior of the system. It was found that moving-platform pose estimation by integration of control output is the most robust and best adapted for different tasks, including the tracking of a mobile object with unknown velocity.

It was found that CDPRs with visual servo control are very robust to many different perturbations, including errors in the CDPR model. To evaluate this robustness, the Lyapunov stability analysis was performed for a planar and a spatial CDPRs. During

the analysis, a link between the moving-platform pose and the system stability was determined. Indeed, in the presence of the same perturbations in the system, it could be stable with the moving-platform on one pose, yet unstable in another pose. Thus, a novel workspace named Control Stability Workspace (CSW) was defined. It shows all the moving-platform poses for which the system remains stable as long as the perturbations are kept within their defined bounds. The workspace had to be computed separately for each visual servoing approach on each of the CDPRs, namely one planar and two spatial ones. By computing the CSW it was possible to quantify the effect of each perturbation on the system. In fact, depending on the perturbed parameter, the effect on the CSW volume is very different. For example, the camera position on the moving-platform could be simply unknown, defining an arbitrary position within the moving-platform geometry. A perturbation on the camera position that does not surpass the moving-platform size has very little to no effect on the CSW volume. On the other hand, errors in cable exit and anchor point coordinates rapidly reduce the CSW volume. Furthermore, the perturbation limit that leaves a reasonable workspace size depends on the CDPR geometry. Indeed, the larger the CDPR, the larger the perturbation within system stability.

As long as the system remains stable, the CDPR reaches the desired state with the same accuracy, regardless of the perturbations. On the contrary, the trajectory to the desired state can have large deviations once perturbations are introduced into the system. An uncontrolled and unpredictable trajectory can be undesirable and even dangerous, especially if the workspace is cluttered. To deal with this issue, trajectory planning and tracking [MC02] has been implemented. A control algorithm was designed, defining the planning and the tracking stages. It was shown that having trajectory planning and tracking guarantees that the executed trajectory is very close to the ideal one, no matter the perturbations. The algorithm was tested on both spatial CDPRs, improving greatly their behavior.

Regardless of the chosen visual servoing approach, some cables can become slack. This can happen for reasons such as modeling errors, accumulation of errors during moving-platform pose estimation, higher number of cables than the number of degrees of freedom of the moving-platform. For this reason new control approaches that take into account cable tensions were proposed. The simplest one consists of a visual servoing controller with a tension correction algorithm. The algorithm compares the current tension measurement to a predefined tension threshold. If the measurement is too low, then a cable velocity correction is computed for that cable to ensure that the slack is reduced. The controller has been validated on two CDPRs showing a substantial difference in robot behavior and system stability compared to the visual servoing approaches without this enhancement. For example, in a long-term task, where the robot is tasked with tracking a mobile object, the classic visual servoing controller fails in less than ten minutes, while the controller with tension correction was fully functional even after three hours.

Organization of the Thesis

The thesis is organized as follows:

Chapter 1 presents the context of this thesis and a review of the pertinent state of the art. More precisely, CDPRs are introduced, giving details on their classification, advantages, drawbacks, and currently known applications. Then state of the art of CDPR modeling, workspace definition, control approaches and stability determination is presented. This chapter also introduces geometric and kinematic models of CDPRs, as well as the computation of the Static Feasible Workspace (SFW) and the Tension Distribution Algorithm (TDA). The last section is dedicated to the introduction of different visual servoing approaches.

Chapter 2 presents different moving-platform pose estimation methods. Altogether seven methods are developed and tested on two CDPRs of different size. Moreover, a comparison is made with the online measurement of the moving-platform pose obtained by an HTC Vive tracker. The moving-platform pose estimation methods are tested with two controllers: an open-loop velocity controller and a closed-loop visual servoing controller. It is found that moving-platform pose estimation by the integration of control output gives the best results when used with the visual servoing controller and is the most robust for several tasks. The accuracy and repeatability of the two controllers are assessed. The findings are submitted for publication in [VI].

Chapter 3 is dedicated the application of visual servoing to CDPRs with a particular focus on stability. First, a thorough Lyapunov stability analysis is done for the PBVS of a planar and a spatial CDPR. A link between moving-platform pose and system stability is discovered. For this reason a new workspace, named Control Stability Workspace, is defined and computed for the different CDPRs and visual servoing approaches. The system behavior in the presence of perturbations is extensively tested with PBVS control both in simulation and on real CDPRs. A short validation of 2½D VS controller is also performed, as well as IBVS. It is concluded that visual servoing of CDPRs is very robust to diverse large perturbations, including modeling errors. The accuracy at desired state always remains the same as long as the system is stable. However, perturbations in the system that are within the bounds of stability can heavily affect the produced trajectory. Furthermore, cable tensions are not controlled and thus cable slack can occur, especially in the presence of perturbations. The findings are published in [II],[V] and [IV].

Chapter 4 presents an algorithm for trajectory planning and tracking to improve the robot behavior in the presence of perturbations. An extensive study is done on ACROBOT to validate the improvement thanks to trajectory planning and tracking. It is also found that the direction of the perturbation matters. More precisely, a perturbation on one system parameter of the same size but different direction will have a different effect on the executed trajectory. While trajectory planning and tracking greatly improves the produced

trajectories, cables slackness occurs nonetheless. Concept is also validated on CAROCA. The results are published in [I].

Chapter 5 proposes some approaches to combine visual servoing with tension management. The Tension Correction Algorithm (TCA) is defined. It compares the current tension measurement to a threshold and computes a cable velocity correction to reduce cable slack. It is validated on both CDPRs with both static and moving objects. The use of TCA on ACROBOT is published in [III].

General Conclusions, as the name suggests contain the main conclusions drawn during this thesis along with propositions for future work. In short, it is concluded that, indeed, visual servoing is an excellent control approach to improve the accuracy of CDPRs. It is very robust to perturbations in the system and modeling errors. Moreover, having perturbations in the system does not affect the accuracy of the robot at the desired state. Indeed, only the transient phase is affected. Here, the robot behavior is greatly improved by using planning and tracking of trajectory. Finally, cable slackness can be an issue, however the proposed TCA removes slack rapidly and ensures that all cables of the CDPR are tensed enough to participate in the displacement of the moving-platform.

Appendix presents: A.1 the two CDPRs located at IRT Jules Verne that were used during this thesis as well as their simulation in V-REP; A.2 the measurement system Creaform C-Track used to obtain ground-truth measurements of the moving-platform pose; A.3 the expression of the equations necessary for the control scheme in a different reference frame; A.4 a simple position controller; A.5 Control Stability Workspace results.

List of Publications

Published:

- Journal articles:
  - [I] Z. Zake, F. Chaumette, N. Pedemonte, and S. Caro, "Robust 2 1/2D Visual Servoing of a Cable-Driven Parallel Robot Thanks to Trajectory Tracking", in *IEEE Robotics and Automation Letters (RA-L)*, Vol. 5(2), pp. 660–667, 2020.
  - [II] Z. Zake, F. Chaumette, N. Pedemonte, and S. Caro, "Vision-Based Control and Stability Analysis of a Cable-Driven Parallel Robot", in *IEEE Robotics and Automation Letters (RA-L)*, Vol. 4(2), pp. 1029–1036, with a presentation in *International Conference on Robotics and Automation (ICRA2019)*, 2019.
- International Conference Proceedings:
  - [III] Z. Zake, F. Chaumette, N. Pedemonte, and S. Caro, "Visual Servoing of Cable-Driven Parallel Robots with Tension Management", in *IEEE International Conference on Robotics and Automation (ICRA2021)*, 2021.
  - [IV] Z. Zake, F. Chaumette, N. Pedemonte, and S. Caro, "Control Stability Workspace", in *Cable-Driven Parallel Robots. CableCon 2021*, 2021.

[V] Z. Zake, S. Caro, A. Suarez Roos, F. Chaumette, and N. Pedemonte, "Stability Analysis of Pose-Based Visual Servoing Control of Cable-Driven Parallel Robots", in Pott A., Bruckmann T. (eds) *Cable-Driven Parallel Robots. CableCon 2019*. Mechanisms and Machine Science, vol 74. Springer, Cham, pp. 73–84, 2019.

Submitted for publication:

- International Conference Proceedings:

  [VI] Z. Zake, F. Chaumette, N. Pedemonte, and S. Caro, "Moving-Platform Pose Estimation for Cable-Driven Parallel Robots", in *IEEE International Conference on Intelligent Robots and Systems (IROS2021)*, 2021, submitted.

1. Background

Contents

1.1 Cable-Driven Parallel Robots

When the Degrees of Freedom (DoF) of an end-effector can be controlled via a mechanical system, this system is called a robot [Mer06]. Two types of manipulators can be distinguished: a serial and a parallel manipulator, shown in Figs. 1.1a and 1.1b, respectively. The former is a series of rigid links connected by motor-actuated joints that extend from a base to the end-effector. In a way, it resembles a human arm. In a parallel manipulator multiple legs connect the base to the end-effector thus creating one or several closed loops. Each leg is either serial kinematic chain or a chain containing one or several closed-loops. One of the first parallel robots was created by Gough in 1954 and is shown in Fig. 1.2, and now this concept has been used countless times, for example, in flight or driving simulation (see Fig. 1.3) [Che+01].

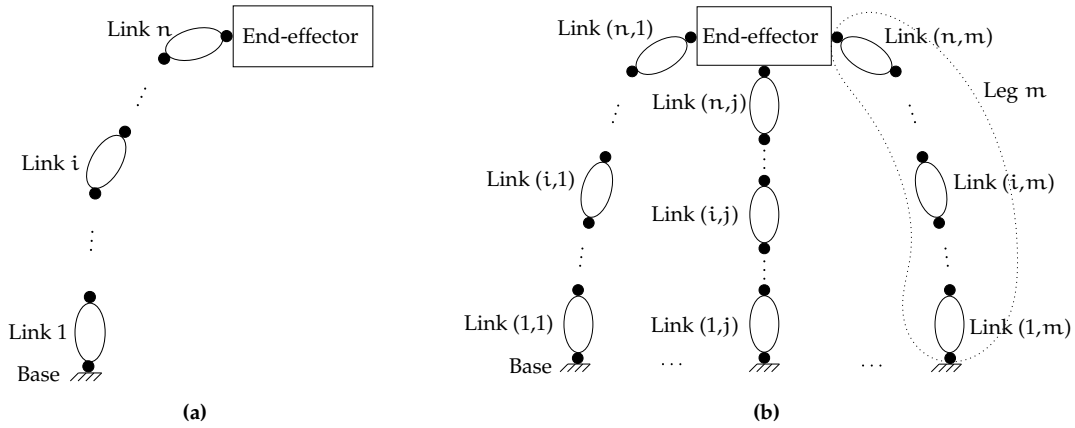


Figure 1.1: Serial and parallel manipulators [Pic18]

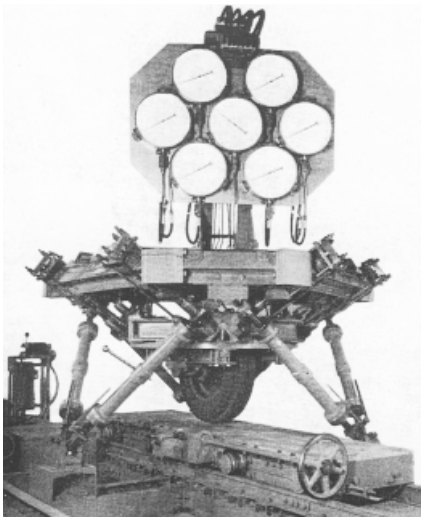


Figure 1.2: The original Gough-Stewart platform

Cable-driven parallel robots (CDPRs) are a type of parallel manipulators, where the rigid links are replaced by flexible cables that actuate the moving-platform (the end-effector of parallel manipulators is usually named moving-platform). On one end each cable is

Figure 1.3: National Advanced Driving Simulator (NADS) at University of Iowa [Che+01]

connected to the moving-platform, while on the other end it is wound on a winch that is actuated by a motor, which itself is rigidly fixed to the base and does not move. Note that the motors and winches do not necessarily have to be fixed to the base, they can also be mounted on the moving-platform for some specific implementations [Ver] [Web16]. Passive pulleys are usually used to guide the cable from the winch to the cable exit point. Generally, the base is fixed to the ground, however there have been studies on mobile CDPRs [Ras+18]. The pose of the moving-platform is controlled by changing cable lengths.

1.1.1 Classification

A CDPR can be classified by its DoF, where the maximum is a six-DoF configuration as shown in Fig. 1.4.

In the scope of this thesis we are interested in CDPRs with six DoF, for which two configurations are possible: either fully-constrained or suspended as shown in Fig. 1.5. For the former the cables are kept in tension by other cables, which means that in theory such a robot could work in a reduced or zero gravity environment. Furthermore, fully-constrained CDPRs are well suited for very fast motions [Kaw+97], however the load capacity is reduced and there is a higher probability of cable collision with objects in the workspace. On the other hand, in suspended CDPRs all cable exit points are above the moving-platform and thus all cables can participate in load bearing. This leads to the ability to lift heavy loads, but the velocities must be reduced to avoid the swaying of the load.

To be able to actuate n-DoF, the CDPR must have at least $n+1$ cables [KI93]. Finally, it is also possible to distinguish CDPRs as a function of degrees of redundancy:

- underconstrained: a CDPR with $m < n+1$ cables for the actuation of n-DoF
- completely constrained: a CDPR with exactly $m = n+1$ cables for the actuation of n-DoF
- overconstrained or redundantly constrained: a CDPR with $m > n+1$ cables for the actuation of n-DoF

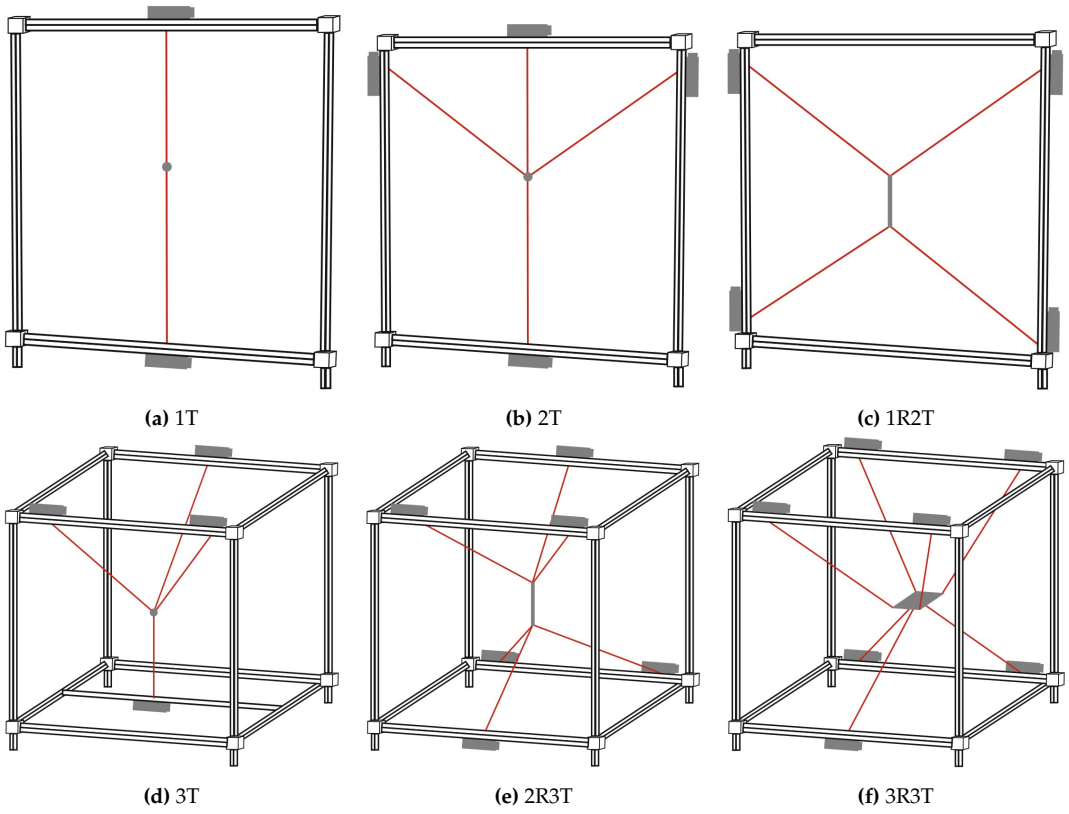


Figure 1.4: Different degrees of freedom achievable with actuation by cables, where T - translational degree of freedom, R - rotational degree of freedom [Pot18]

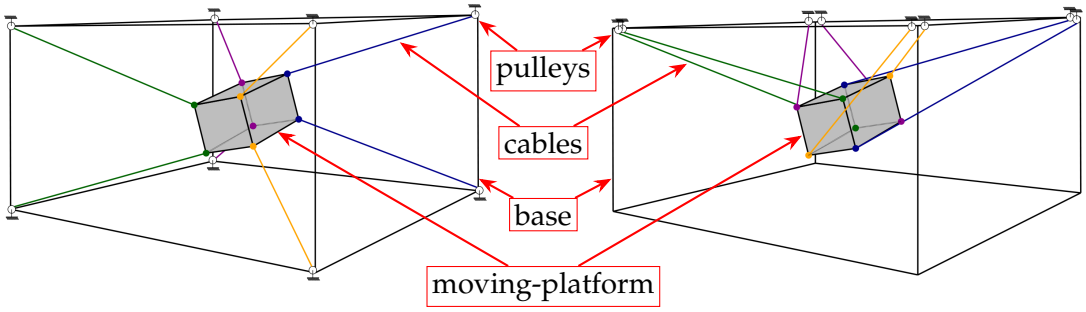


Figure 1.5: A spatial CDPR with six DoF. Left: Fully-constrained configuration. Right: Suspended configuration.

In practice, most of the CDPRs are redundantly constrained with eight cables to actuate six-DoF, making their degree of redundancy r equal to 2.

1.1.2 Advantages and Drawbacks

The choice to use cables instead of rigid links leads to many advantages, such as low mass of the moving parts, because only cables and the moving-platform are in motion. Indeed, generally the motors are fixed on the base (on the ground) and thus they do not move. Furthermore, cables are considerably less heavy when compared to rigid links of classic parallel robots, such as the Gough-Stewart platform in Fig. 1.2.

Consequently, having low mass in motion gives the possibility of achieving high velocities and accelerations of the moving-platform, especially if the CDPR is assembled in the fully-constrained configuration. For example, a CDPR named SEGESTA can achieve up to 10g acceleration and velocities around 10 m/s [Bru+06].

As the cables are wound on winches, there is generally no limit on available cable length. This leads to great scalability of CDPRs, allowing for different CDPR workspace sizes.

A CDPR can be easily reconfigured [Iza+13] [Gag+16]. That is, the pulleys working as cable exit points on the base can be displaced to change CDPR geometry, an example is shown in Fig. 1.6a[1]. Similarly, the shape of the moving-platform can be adapted to the task at hand and to the sensors that need to be placed on it.

CDPRs have a large payload capacity, especially if they are assembled in the suspended configuration. When compared to an overhead crane, which is typically installed in almost every factory, a CDPR could keep the load more stiff and thus providing better safety. Moreover, all six DoF of CDPR moving-platform can be controlled, thus increasing their application range.

It should be noted that CDPRs can be portable and deployable [MD10] [BWT05], as shown in Fig. 1.6b. There have even been studies on mobile CDPRs, whose frame is mounted on two or four mobile bases [Ras+18] [Ras19].

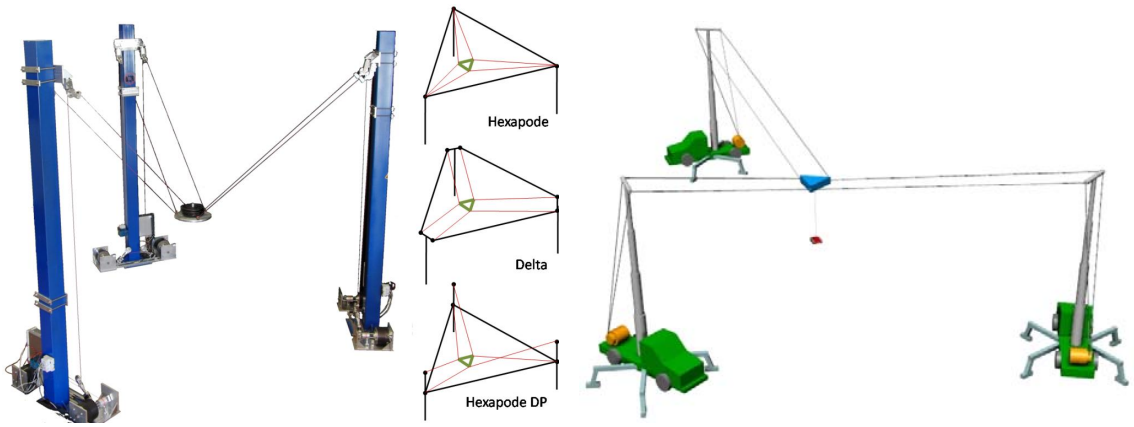


Figure 1.6: An example of **(a)** a reconfigurable and **(b)** a deployable CDPR structure [BWT05]

However, the use of cables instead of rigid links has some drawbacks too. As mentioned above, CDPRs are usually redundant, with more cables than degrees of freedom to compensate the fact that each cable can apply only a unidirectional force, that is, it can only pull but it cannot push. At least $m = n + 1$ cables are needed to fully constrain n degrees of freedom (DoF) [RGL98], making a CDPR redundant in actuation to degree

[1] Dynamic reconfiguration of a CDPR can be seen in `https://youtu.be/ja98QlLI5gc`

$r = m - n$. Generally, eight cables are used for six DoF motion, because this increases the CDPR workspace [LG13] [Gou+15].

Cables are elastic, they can sag under their own weight and become slack during task execution. All of this significantly complicates the cable geometry [Irv92], thus impacting the CDPR model and control. Usually a simplified model is chosen, assuming that cables are massless, inelastic and straight [Gag+16] [Pic18]. This however can have a negative impact on CDPR accuracy and stiffness [Sch17]. The loss of moving-platform stiffness is directly correlated with increase in cable slackness. Indeed, if some cables become slack for example due to CDPR model imprecision, then the moving-platform loses its stiffness. In more extreme cases, cable slackness can lead to underactuation of the moving-platform. On the other hand, it is possible to model cable elasticity and sagging [MADS15] [Sch17] [Web16]. While that will not prevent cable slack, it can extend the time necessary to accumulate a significant amount of slack. However, often additional sensors, such as cable tension and cable angle sensors, will be necessary in order to use more complex cable models [Dal+19] [FCCCL16]. A compromise between model complexity and system behavior needs to be found for each individual task and each CDPR geometry.

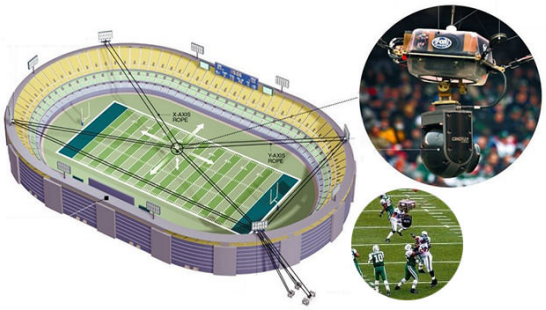
Even without cable modeling issues, the geometry of CDPRs is complex. Indeed, CDPRs are part of parallel robots and thus the solution of the Direct Geometrico-Static Model (DGSM) is a challenging issue, because many solutions can exist. Furthermore, due to the fact that cables can only exert one-directional force on the moving-platform, the complexity for CDPRs is even higher [CM13] [AC15]. Usually interval analysis is used to solve DGSM [BMC16]. This is possible even for complex cable models [MADS15], however it comes at a high computational cost.

CDPR payload capacity is generally lower than that of an overhead crane, e.g. a typical maximum is 1 ton [Pot18]. Furthermore, payload capacity depends on the pose of the moving-platform. More precisely, the higher the moving-platform along global z axis and the closer the cables to being horizontal, the higher the cable tensions. Indeed, cable tensions can become infinite in such configurations, which imposes limitations on the workspace of CDPRs.

1.1.3 CDPR Applications

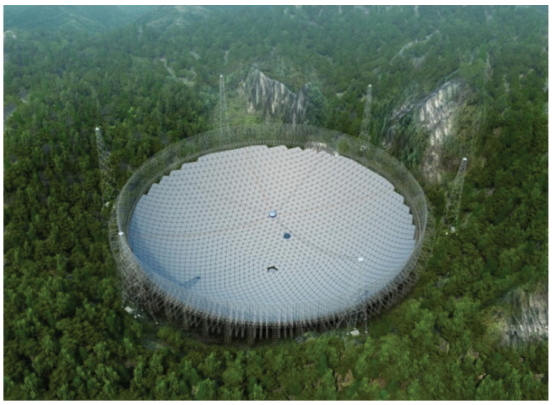
Possibly the best known commercial CDPR application is the cable-suspended camera that is used to broadcast an immersive view of a sport or entertainment event. One such example is SkyCam [Sky], shown in Figs. 1.7a and 1.7b, but there are several others, such as CableCam and SpiderCam. The structure is always the same: four cables are used to actuate the translational DoF of the moving-platform, where the camera is mounted. The rotational DoF are actuated separately and usually the camera is moved by teleoperation. A CDPR is used for this application, because it can easily cover an arena or a football field, while not obstructing the view since cables are rather small in diameter.

(a) SkyCam



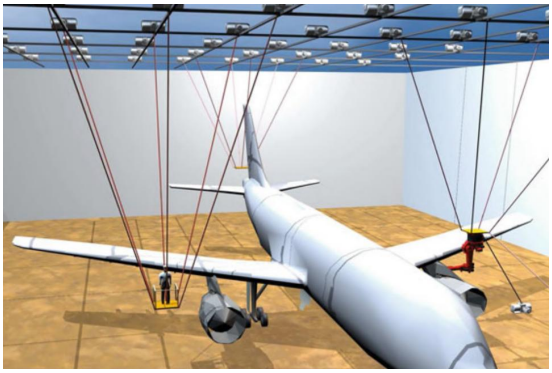
(b) SkyCam attachment on the arena

(c) A plane secured by a CDPR in a wind tunnel

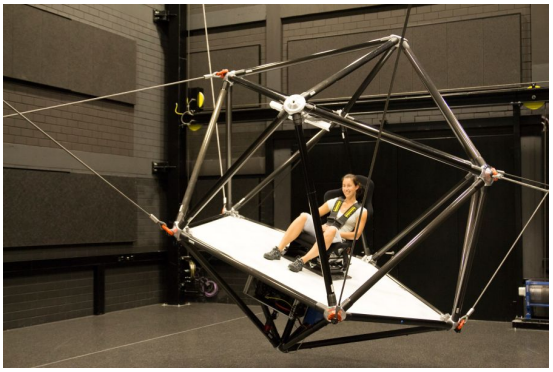


(d) FAST telescope

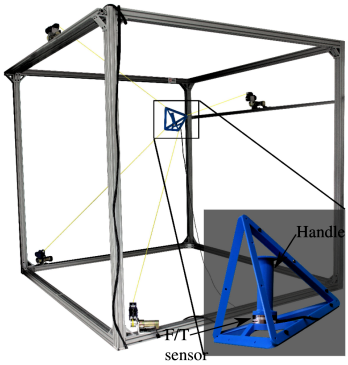
(e) NIST RoboCrane [ABD92]



(f) CableBOT [Cab]



(g) CableRobot Simulator [Mie+16]



(h) haptic interface at Laval University [FCCG14]

Figure 1.7: CDPR application examples

Similarly, CDPRs are very convenient for wind tunnels [LLR02] [BSW11] [Yan+10]. Any object can be secured in the wind tunnel with cables whose diameter is small enough to not cause any unwanted turbulence. Furthermore, it is even possible to change the orientation of the object during a test, which creates a more realistic result. Indeed, planes and cars change course with respect to the wind direction, thus ideally this change needs to be simulated as well. An example is shown in Fig. 1.7c.

Probably the largest CDPR in the world is Five Hundred Meter Aperture Spherical Telescope (FAST) [NL11], shown in Fig. 1.7d. Here, the pose of the receiver is changed via six very long cables. Furthermore, the receiver weighs 50 tons, meaning that the cables used to support it are large in diameter, have non-negligible weight and, evidently, cable sag needs to be taken into account.

Large workspace capability is also interesting in indoor applications. Tedious and monotonous tasks, such as cleaning and painting of airplanes can easily be done by CDPRs [ABD92] [NG14] [Cab], as shown in Figs. 1.7e and 1.7f.

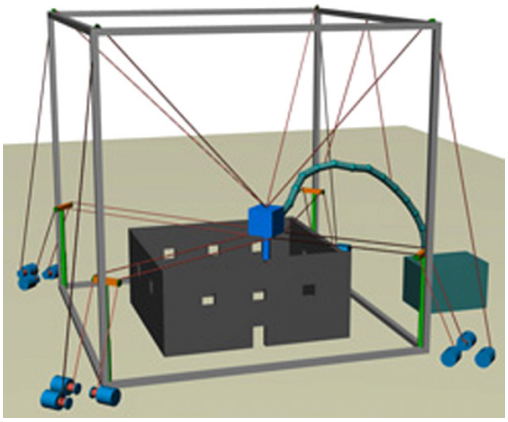
Similarly to the Gough-Stewart platform, a CDPR can be used in motion simulation. CableRobot Simulator [Mie+16] has been built at Franhofer IPA in Stuttgart, Germany and is shown in Fig. 1.7g. Unlike the Gough-Stewart platform, a CDPR can produce not only rotational motion, but also a rather large translational displacement. This allows for an immersive sensory feedback to a virtual reality headset.

When using a virtual reality headset it is surprisingly easy to believe what the eyes see. However, it is more difficult to keep this sense of reality once some physical interaction needs to happen. For example, if an object is picked up and is seen in the hand, one would expect to feel it by touch and to sense its weight. Haptic interfaces are created to give this physical feedback. A significant amount of research has been concentrated on the use of CDPRs as haptic interfaces [DPL07] [YY09] [Ho+14] [FCCG14]. An example is shown in Fig. 1.7h. Such haptic interfaces allow the operator to test the design of a part, an assembly or even the whole factory layout before manufacturing has begun.

As mentioned previously, CDPRs can be deployable and even mobile. The latter can be useful in storage facilities, especially if objects are stored on high shelves, as shown in Fig. 1.8c. In other cases a deployable CDPR can be preferred, for example, when the load to lift is rather large or when the CDPR would need to stay at the same location for a significant amount of time. Considering the simplicity of CDPR transportation and setup, it can be used for rescue operations [MD10] and for contour crafting [Bos+07]. The examples are shown in Figs. 1.8a and 1.8b.

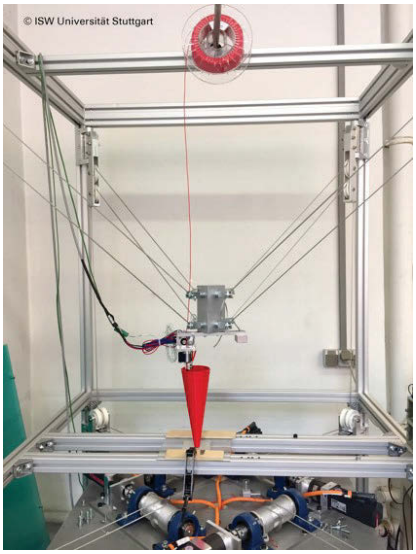
Similarly, in [Iza+17] walls were printed by a CDPR, however in this case it was indoors, as CDPRs with a well known unchanging geometry are bound to be more precise. In both applications, automation brings an increase in speed. An on-site printer could finish the walls of a house in one day. And thanks to its large workspace, this could be done without moving the base of the CDPR and thus no or just minimal intervention would be

(a) MARIONET for patient transportation [MD10]

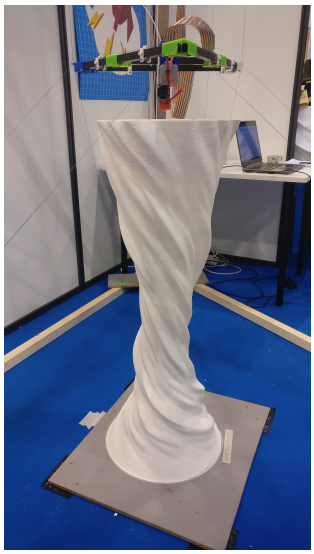


(b) A portable CDPR for contour crafting [Bos+07]

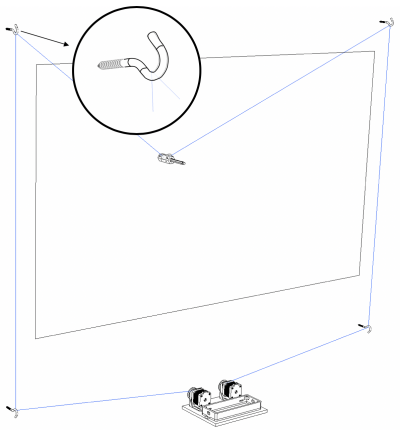
(c) FastKit, a mobile CDPR [Ras19]



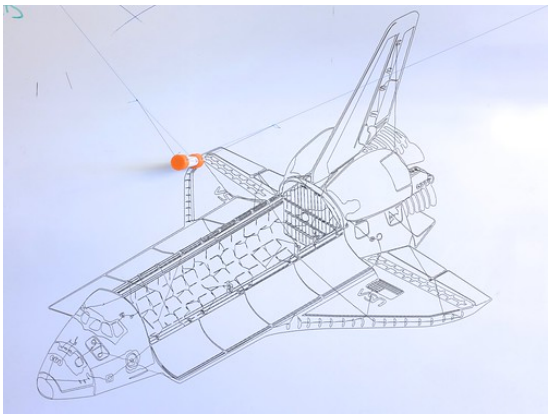
(d) the CaRo printer [Pot+19]



(e) the HangPrinter [Lud]



(f) a schematic of a wall plotter [PVF20]



(g) drawing created by a wall plotter [Pol]

Figure 1.8: CDPR applications, part two

necessary. On the other hand, a set of printed pieces could be easily assembled in short time on place. And since in this case no CDPR on-site setup would be necessary, it is possible that the overall time spent on building the house would be the same.

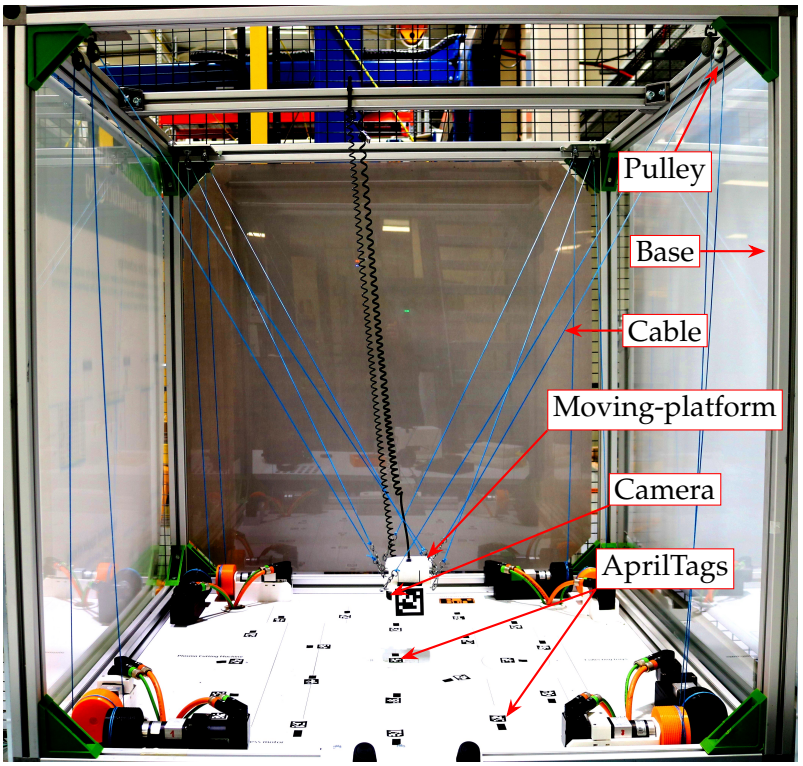
Rapid prototyping, also known as 3D printing, of (usually plastic) pieces is also a viable application of CDPRs. The authors of [Pot+19] proposed a suspended CDPR, named CaRo printer, shown in Fig. 1.8d, and defined the workspace, where the 3D printing accuracy requirements were fulfilled. Furthermore, the authors showed great long-term running robustness as the robot was exhibited in an exhibition during six months. Another interesting CDPR 3D printer, named HangPrinter, was created by Torbjørn Ludvigsen and is shown in Fig. 1.8e. This is an open-source project, that is available at [Lud]. In his approach the printer uses a set of parallel cables coming from the top plate to control the height of the moving-platform along z axis. Three other cable pairs are coming from the ground and are used to control the displacement on xy plane. The author claims that this printer only needs the top plate to be rigidly mounted on the ceiling and the three ground points to be well anchored for it to work, meaning that theoretically there is no size limit, as shown in [Han].

Indeed, CDPRs can be very interesting as autonomous creators of art pieces, especially because the cables can be chosen so that they are not clearly visible, giving the impression that the moving-platform is actually floating in the air on its own. A new trend is emerging in using very simple two-cable CDPRs as wall plotters [Pol] [PVF20], shown in Fig. 1.8f. These wall plotters can be set up on any wall and can create large and detailed art pieces, such as the one shown in Fig 1.8g. The wall plotters are so popular and simple, that they have become commercially available [Scr]. A recent paper [PVF20] describes the study of exhibition visitor reactions to CDPR art and the ability to interact with the robot and the art piece by the push of a button.

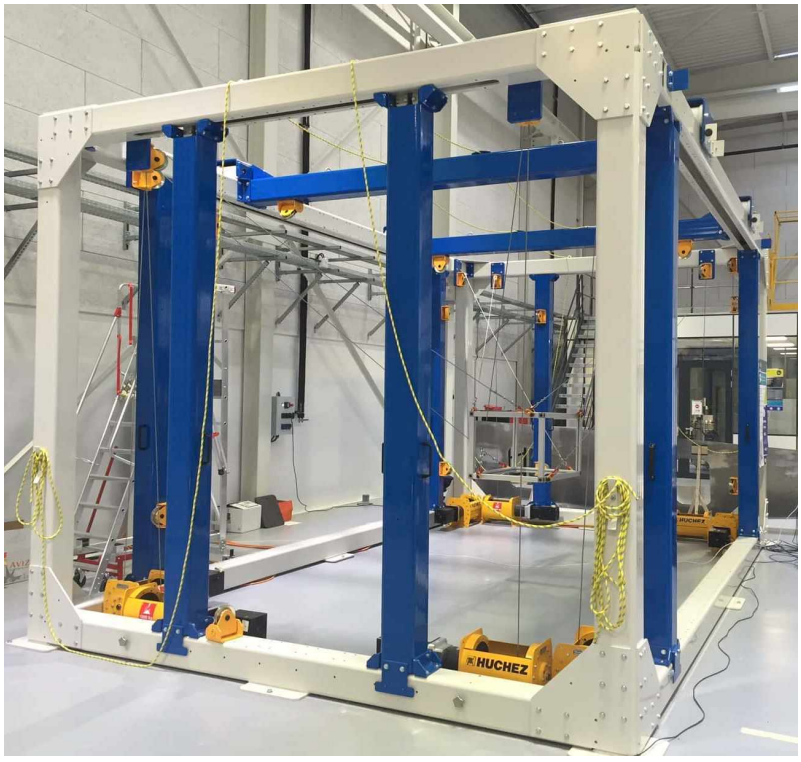
Clearly, the applications of CDPRs are vast and very different, ranging from industrial use to creation of art pieces.

1.1.4 CDPRs at IRT Jules Verne

Two CDPRs were used during this thesis: a small CDPR prototype named ACROBOT and a large CDPR named CAROCA. ACROBOT was developed as a demonstration prototype that can be transported to exhibitions thanks to its compact size of $1.2\,\mathrm{m} \times 1.2\,\mathrm{m} \times 1.2\,\mathrm{m}$ cube. It is shown in Fig. 1.9a. On the other hand, CAROCA has the following dimensions: $7\,\mathrm{m} \times 4\,\mathrm{m} \times 3\,\mathrm{m}$. Furthermore, it is reconfigurable and its different reconfiguration strategies were studied in [Gag16]. CAROCA was developed to research the possibility of CDPR industrialization for tasks such as photogrammetry, sandblasting and painting of large complex parts, handling operations [Pic18]. CAROCA is shown in Fig. 1.9b. Detailed information on both CDPRs, including their actuation and sensors, is given in Appendix A.1.



(a)



(b)

Figure 1.9: CDPR prototypes at IRT Jules Verne: **(a)** ACROBOT; **(b)** CAROCA

1.2 State of the Art

One of the earliest studies on CDPRs was carried out for naval applications in the early 1990s. Four cranes were used to load the containers from boats to docks and vice versa [GJC01]. At the same time the NIST project, shown in Fig. 1.7e began.

By the end of 1990s the amount of research focused on CDPRs increased considerably. For example, a joint project in Canada and the United States was carried out on the characterization of the workspace in which a set of desired tensions could be generated by the cables [RGL98]. This is a fundamental issue, because the cables can apply only a one-directional force. Later on, notable work on deployable CDPRs for rescue missions was done [MD10], shown in Fig. 1.8a. An elasto-static model of the ANR CoGiRo prototype was developed to take into account the prototype's heavy cables [Lam14]. Furthermore, the cited work also proposed a very interesting arrangement of the cables on the moving-platform to improve the robot behavior when in the suspended configuration.

More recently, CDPR reconfigurability has been studied in [Gag+16] [Iza+13]. Indeed, as the moving-platform is actuated by cables, it is possible to change the robot geometry and behavior by simply changing the cable exit point coordinates.

In the following sections, the state of the art on the relevant research work on CDPRs is presented.

1.2.1 Modeling

The modeling of CDPRs is similar to that of conventional "rigid" parallel robots, the main difference being the modeling of the cables. In the majority of existing work, the cable is considered to be of negligible mass. Its longitudinal elasticity is also often neglected. However, there are works on the quasi-static analysis of parallel robots driven by ropes whose mass and elasticity are not negligible, e.g. [Gou+12] and [KZW06]. In [Sch17] Schmidt analyzed the importance of every CDPR model improvement by measuring the resulting accuracy. It was found that cable elasticity and pulley geometry are the most influential parameters. The robustness of sensor control techniques generally allows very simplified models to be considered without loss of accuracy or stability. In the framework of this PhD thesis, the simplest CDPR model is used and the influence of the unmodeled characteristics, such as pulley kinematics and cable slackness, is analyzed.

A second noteworthy aspect of CDPR modeling is the solution of the direct geometrico-static model (DGSM), where the aim is to determine the moving-platform pose(s) and the corresponding tensions knowing the cable lengths. For all parallel robots the DGSM is a challenging issue, because many solutions can exist. However, for CDPRs the complexity is higher due to one-directional force that the cable can exert on the moving-platform. The complexity of the problem has been highlighted by Carricato *et al.* for suspended CDPRs with less than six cables in [CM13]. More precisely, for a 3-cable CDPR the solution of DGSM is equivalent to finding the roots of a univariate polynomial

of degree 156 [Car13] and of degree 216 for a 4-cable CDPR [CA13]. Finally, in 2015 the analysis was extended to a generic underconstrained CDPR with n cables [AC15].

The use of interval analysis in the DGSM has proven to be one of the most efficient methods [Ber15] [BMC16]. The proposed algorithm can be applied to both the fully constrained and the suspended CDPR and it provides the complete solution of the DGSM with all the moving-platform poses that could be reached by the given cable length set. Furthermore, interval analysis can also be used for more complex CDPR models, for example with the sagging cable model [MADS15]. However, it should be noted that finding all the solutions has a high computational cost.

Once all solutions are available, it is however necessary to determine which is the most probable one. Collard and Cardou propose to find the moving-platform pose with the lowest equilibrium [CC13]. Indeed, for suspended CDPRs the most likely moving-platform pose is the one where the moving-platform center of mass is closest to the ground, which corresponds to the lowest equilibrium pose. The efficiency of the algorithm is proportional to the amount of cables of the suspended CDPR. A common work-around is to start at a known moving-platform pose and use an iterative scheme to track that solution [Fan05].

1.2.2 Workspace

A workspace consists of all the CDPR moving-platform poses, in which certain conditions are fulfilled. Depending on the selected conditions multiple workspaces can be computed.

The simplest one is the Static Feasible Workspace (SFW), where the static equilibrium of the moving-platform is used as the condition [SK06] [Gag16]. The Wrench Feasible Workspace (WFW) is based on the static equilibrium along with the condition that the moving-platform is able to withstand a set of external wrenches, named Required Wrench Set (RWS), without moving. By definition, the WFW is the set of wrench feasible moving-platform poses. A pose is wrench feasible when the cables can balance a set of external moving-platform wrenches while the cable tensions stay in between given cable tension upper and lower bounds [BREU06] [GDM11]. It can be said that SFW is a special case of WFW when the external wrench is equal to zero and the static equilibrium is achieved by compensating only the moving-platform weight. The tension bounds τ_{ub} and τ_{lb} are defined so that the tension does not surpass the cable or CDPR structure breaking point on the upper bound and so that the cables cannot become slack on the lower bound.

Cable-cable interference can affect the pose the moving-platform attains. Indeed, as the cable is no longer a straight line, but instead is bent at the interference point, the model no longer corresponds to the real CDPR and thus a different moving-platform pose is obtained. To avoid such situations one can compute the Interference Free Workspace (IFW), which is the set of all moving-platform poses without collisions between cables [Per+10].

An extension of this workspace is the Collision Free Workspace (CFW), which is the set of all moving-platform poses without collisions between the cables, the moving-platform, and the environment.

For some tasks a CDPR will be required to attain high velocities and the dynamics of the robot will need to be taken into account. In such a case SFW and WFW cannot be used, instead the dynamic workspace needs to be considered. For example, a Dynamic Feasible Workspace (DFW) is the set of dynamic feasible moving-platform poses. Here, a pose is dynamic feasible if the prescribed moving-platform acceleration set is feasible, while cable tensions are kept within their bounds. For the planar CDPRs the dynamic equations were solved analytically and thus the boundary of DFW could be found [BG05]. However, the same strategy cannot be applied to spatial CDPRs due to their increased complexity. Kozlov in [Koz14] extended the method for WFW computation described in [Gua+13] so that DFW can be computed. However, this method does not take into account dynamics of cables and winches, nor the external wrenches acting on the moving-platform. For this reason Gagliardini proposed a new workspace called Improved DFW in [Gag16] [GGC18].

All of the mentioned workspaces portray the physical ability of the moving-platform to arrive at a certain pose given a set of conditions that need to be fulfilled. The control of the CDPR is not taken into account, meaning that there is no knowledge whether the controller will be able to guide the moving-platform to the desired pose.

1.2.3 Control

Most of the existing control strategies for CDPRs are based on well-known methods, such as PID controllers. Most CDPRs are a part of the family of over-actuated systems, i.e. they have more actuators than degrees of freedom. This is motivated by the following fact: the cables constituting these parallel robots can only pull and not push on the platform. The presence of additional actuators is then a necessary condition for the control of all the degrees of freedom of the robot [LG13] [Rui+15] [Pot18]. In addition, the presence of redundant actuators increases the space that can be reached by the robot [LG13]. Thus, recent works in the literature focus mainly on how to control the cables that actuate the moving-platform while ensuring :

- a positive tension at all times for all the cables;
- a time-continuous tensioning, for safety reasons but also for equipment wear and tear;
- to solve this control problem under strong real-time implementation constraints.

In general, at least the first two points are addressed. Most often a classic control approach, such as torque control or sliding mode [OA04] [Hu+14], is used and the tension distribution is taken into account by solving an optimization problem, which minimizes the energy cost [Lam14] [Bor+09]. If the trajectory generation must be done online, it will imply solving an optimization problem at each iteration, which can be difficult with the

real-time constraint. For this reason, some approaches propose the use of simplifying heuristics [Bor+09] [Mik+08].

For some CDPRs dynamic model must be taken into account in the control, otherwise the system can exhibit some unsatisfactory behaviors. For example, cable elasticity is taken into account in [Rie11] [Sch17] [Pic18]. Another parameter that can vary strongly in the model is the embedded mass [Pic+18b].

Modeling all the cable characteristics can be a tedious task, which can become close to impossible if the control needs to be recomputed at every iteration. Schmidt proposed in [Sch+17] to use a simple CDPR model and to record the accuracy with which each pose in the workspace can be attained. The authors first divided the workspace of their CDPR into a grid of 1920 points. Each point was visited by the moving-platform and its pose was measured with a Laser tracker AT901-MR with precision of 5 µm and the difference between the desired and the measured pose was recorded in a distortion grid. Then the desired trajectory was corrected by taking into account the distortion grid. Indeed, since a CDPR has high repeatability, it is possible to adjust the trajectory with the distortion grid before it is to be given to the controller. This leads to the actual executed and measured trajectory corresponding to the original desired one. Three trajectories were tested and a substantial improvement of accuracy was seen in two of them, going from approximately 10.5 mm to 1.65 mm for a 7.0 m $\times$ 4.0 m $\times$ 3.0 m workspace. While the attained accuracy is impressive, the approach is not robust and cannot be easily reused on different CDPRs. Indeed, if the knowledge of initial pose is bad for any reason, the correction algorithm has no way of knowing that and thus the trajectory and the final pose will not be the desired one. Furthermore, the measuring step would need to be repeated in cases such as: changes in moving-platform geometry and cable wear.

In general, to increase the robustness of the controller, some additional sensors are used to provide feedback. Indeed, modeling of the whole environment is an impossible task, because of many uncertainties and unpredictable perturbations. For example, when dealing with considerable loads cable extension is unavoidable. For this reason, Picard used force sensors to measure cable tensions [Pic+18a]. This allowed to determine the actual mass of the moving-platform (with or without load), which noticeably improved the CDPR behavior and final accuracy. Another approach is to use angular position sensors to measure cable angle position [FCCCL16]. Weber proposed a control algorithm with feedback from onboard inertial measurement units (IMUs) that actively damps vibrations of the moving-platform [Web16]. Finally, in several studies, cameras were used to measure the moving-platform pose [Dal+11] [Dal+12] [Beg+18] [Dal+19] [CCL15]. Furthermore, cameras were used to measure the cable angle and compute cable sag [Dal+12] [Dal+19]. It is also possible to use the camera as an exteroceptive sensor such as in [RCM14], where it was mounted on the moving-platform and used to detect the object of interest.

Vision-Based Control of CDPRs

While model-based approaches insist on perfecting the CDPR model in order to have a good accuracy, it is in general very difficult if not impossible to reach a millimetric accuracy. This of course is due to the fact that no matter the CDPR model complexity and closeness to the real robot, this control will not be robust to unmodeled changes or perturbations. Furthermore, increasing the amount of details in the model increases the complexity of computations and the time needed to get a solution.

Adding vision sensors to the system allows the control to be robust to changes in the environment and, more importantly, to modeling errors of the CDPR. When using vision sensors, a simplified CDPR model can be used. This is because the accuracy will be achieved by the vision-system and not by the accuracy of the CDPR model. It also allows us to avoid the computation of the direct geometrico-static model in some cases. Furthermore, a vision-based approach is the most natural in a sense that it corresponds to "eye-based control" in humans. Some possible use cases for a CDPR with vision-based control are: (i) handling small objects over potentially large spaces; (ii) finding the correct object among others; (iii) assembly of large parts in large spaces; (iv) large scale 3D printing with different materials.

Visual servoing (VS) control for CDPRs is yet to be intensively researched. The authors are aware of only a few studies related to this topic, while it should be noted that all of them show promising results. Dallej *et al.* [Dal+11] describe an approach with a camera onlooking the CDPR moving-platform, which is equipped with a marker to simplify its recognition and thus computation of its pose. This choice to externally sense the moving-platform pose in the feedback signal allows to avoid the solving of the forward geometrico-static problem. In experimental tests on a 6-DoF CDPR named ReelAx8 the approach shows a good accuracy. However, the authors conclude that another approach is needed to deal with more dynamic tasks. Authors then move on to a second control algorithm, which takes into account the dynamics of the moving-platform. The controller is a standard computed torque controller in the Cartesian space, namely a PD with a feed-forward, where the camera is used to determine the pose and the velocity of the moving-platform. This controller is validated only in simulation with added noise to the estimation of the moving-platform pose and velocity expressed in the base frame. A further improvement to this algorithm is shown in [Dal+12]. Here, a combination of multiple cameras is used to better determine the pose of the moving-platform. Furthermore, additional set of cameras is used to detect the cables and compute their angles at their exit point, which in turn leads to the determination of cable sag. This is necessary to improve the previously described dynamic model. Now the dynamics of the cables are also taken into account. The validation is done in simulation and shows promising results. Furthermore, tension sensors are also used to measure cable tensions that are necessary for the dynamic sagging cable model with non-negligible mass. The reported accuracy is four times better (10 mm vs. 39.6 mm) and it is validated on a real robot [Dal+19].

The resulting accuracy is especially impressive, when considering the size of the robot: 15 m × 11 m × 6 m.

Similarly, Chellal *et al.* proposed an approach with 6 infra-red cameras to precisely determine the pose of the moving-platform [CCL15]. They used a Bonita motion-capture system by Vicon to measure the moving-platform pose of the 6-DoF CDPR named INCA. The algorithm is rather slow at approximately 6 Hz, but provides a high tracking accuracy of less than 1 mm and less than 1° for a 3 m × 3 m × 3 m workspace. It should be noted that this was designed as a haptic device for a virtual reality feedback.

Ramadour *et al.* take a different approach, by choosing to fix a camera on the moving-platform so that the target object can be directly observed [RCM14]. Indeed, there are no more cameras in the workspace and the control is computed relative to the target object. Furthermore, the authors simplify the task, by using a 3-DoF CDPR with only the translational degrees of freedom. This configuration was achieved by using just 4 cables and connecting them all in a single point on the moving-platform, therefore making the rotational DoFs impossible to control. This approach was chosen, because the moving-platform was equipped with a magnet and the orientation of the object during the pick&place task was not important. Due to the choice of control, the authors still required the computation of the forward kinematic model in order to acquire the current moving-platform pose. This is unlike Dallej *et al.* in [Dal+11] [Dal+12] and [Dal+19], where the cameras were used to measure the moving-platform pose. The approach is experimentally validated on a CDPR named Marionet-Assist [MD10]. The results clearly show that the estimation of the actual moving-platform pose can be rather coarse without making the task impossible. That is, even though the final computed moving-platform pose in the base frame did not correspond to the actual one, it did not affect the system too much, and the pick&place task was successful.

1.2.4 Stability

The stability of CDPR usually deals with the physical stability of the robot. In other words, a moving-platform is stable if it is in a static (or dynamic) equilibrium [Pot18]. A static equilibrium pose is where the moving-platform will arrive once all external wrenches are removed and only gravity is acting on it. For example, Carricato *et al.* investigated the stability of equilibrium for underconstrained CDPRs [CM13]. A CDPR is called underconstrained if the moving-platform can be moved by applying some external forces on it, while the cable lengths are not changed. For example, one could attach moving-platform with only 3 cables. Then the pose of the moving-platform at equilibrium would depend on the location of the center of mass of the moving-platform. However if one would apply a force to the moving-platform, it could be moved while keeping the cables in the same length and still in tension. This analysis allows to determine the region in which the moving-platform can move due to external force, while the lengths of cables stay unchanged. By doing this analysis, the authors were able to determine the

final equilibrium pose, where the moving-platform would return to once all perturbations are ceased. Similar studies are presented in [SR19] underconstrained suspended CDPRs for rehabilitation. Bosscher proposes a slope-based stability measure of a pose of an underconstrained CDPR in [BEU04]. Park *et al.* present a stability analysis method for dynamical CDPRs based on the cable tension bounds [PCM05].

There is a link between moving-platform stiffness and stability. The conditions to analyze the stability of the cable robot are described in [BK06]. In particular, a CDPR can be stabilized in the absence of external load if the kinematic Jacobian matrix is regular and the active stiffness matrix is positive definite.

The mentioned research does not cover the stability of control. Indeed, it is possible that at some moving-platform pose, given a set of other parameters, the system becomes unstable and the control output makes the robot deviate from its target. In control it is common to analyze the Lyapunov stability of the system. It allows us to determine whether the system will converge to its goal state despite errors in its model [Kha02].

1.3 CDPR Modeling

The geometric and kinematic modeling of a six-DoF CDPR with eight cables is described in this section. The most basic model is created, considering cables as straight lines that are mass-less and non-elastic. An extension of the model with the pulley geometry taken into account is also presented.

A planar CDPR is also defined, as it will be necessary for a case study in stability analysis, which is presented in Section 3.2.1. Finally, a very simple position controller is proposed in Section A.4.

1.3.1 Geometric Modeling

The geometric model is the link between motor positions in joint space and moving-platform pose in Cartesian space. The Direct Geometric Model (DGM) expresses the moving-platform pose as a function of motor positions, as shown on the bottom part of Fig. 1.10. The Inverse Geometric Model (IGM), as the name suggests, does the inverse, i.e., it expresses the motor positions as a function of moving-platform pose (Fig. 1.10 top part).

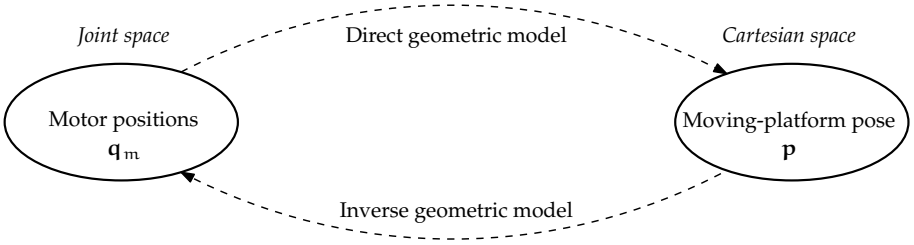


Figure 1.10: Input and output for direct and inverse geometric models

For straight, inelastic and massless cables their length is directly proportional to the motor position:

$$\mathbf{l} = \mathbf{l}_0 + r_w(l)(\mathbf{q}_m - \mathbf{q}_{m0}) \tag{1.1}$$

where

- $\mathbf{l}_0$ is the cable length vector at $t = 0\,s$, that needs to be computed from the moving-platform pose;
- $\mathbf{q}_{m0}$ is the motor position vector at $t = 0\,s$;
- $\mathbf{q}_m$ is the current motor position vector;
- $r_w(l) = r_w + d_c/2$, where r_w is the winch radius and d_c is the cable diameter;
- note that $r_w(l)$ can change with time, if the cable can be wound on a second or nth layer.

For ACROBOT and CAROCA $r_w(l)$ is defined in Appendix A.1.

Basic Inverse Geometric Model

The schematic of a spatial CDPR with 8 cables is shown in Fig. 1.11. Here, A_i denotes the exit point of the ith cable. In the simplest model the pulleys that are actually located at the cable exit points are omitted from the model, thus the cable exit point is assumed to be static and perfectly known. On the other end the ith cable is connected to the moving-platform at its anchor point B_i. The vector $\mathbf{a}_i$, shown in Fig. 1.12, points from the origin of base frame $\mathcal{F}_b$ to the cable exit point A_i. Likewise, the vector $\mathbf{b}_i$ points from the origin of moving-platform frame $\mathcal{F}_p$ to cable anchor point B_i.

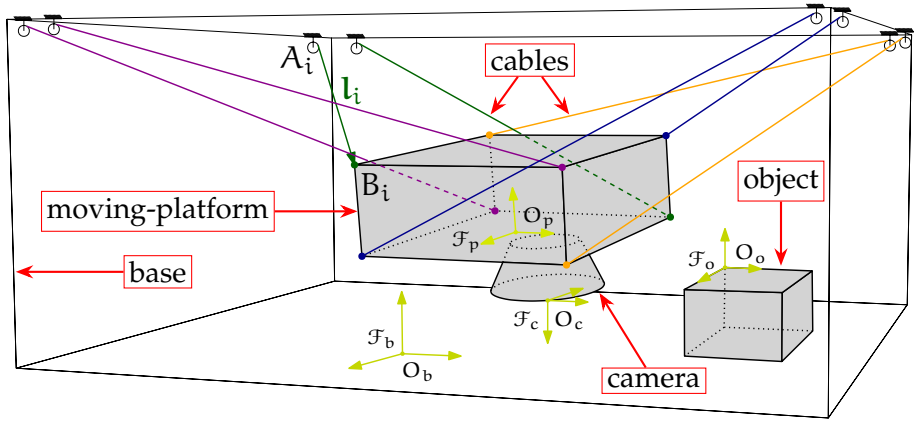


Figure 1.11: Schematic of a spatial CDPR with eight cables, a camera mounted on the moving-platform and an object in the workspace

The length l_i of the ith cable is the 2-norm of the vector $\overrightarrow{A_iB_i}$ pointing from cable exit point A_i to cable anchor point B_i, namely,

$$l_i = \left\| \overrightarrow{A_iB_i} \right\|_2 \tag{1.2}$$

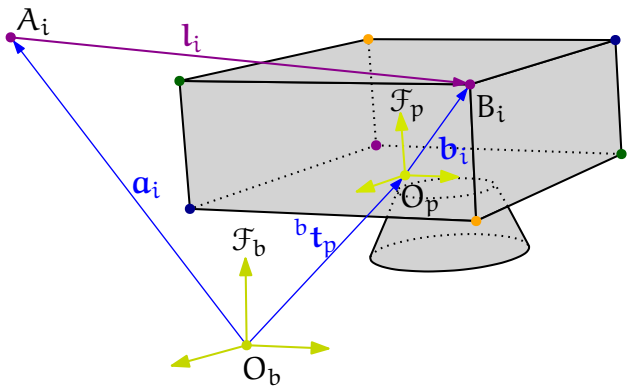


Figure 1.12: CDPR geometric parametrization

Thus the ith cable vector $\mathbf{l}_i$ can be expressed as:

$$\mathbf{l}_i = l_i {}^p\mathbf{u}_i = {}^p\overrightarrow{A_iB_i} = {}^p\mathbf{b}_i - {}^p\mathbf{a}_i = {}^p\mathbf{b}_i - {}^p\mathbf{R}_b {}^b\mathbf{a}_i - {}^p\mathbf{t}_b \tag{1.3}$$

where ${}^p\mathbf{u}_i$ is the unit vector of ${}^p\overrightarrow{A_iB_i}$ that is expressed as:

$${}^p\mathbf{u}_i = \frac{{}^p\overrightarrow{A_iB_i}}{\left\|{}^p\overrightarrow{A_iB_i}\right\|_2} = \frac{{}^p\mathbf{b}_i - {}^p\mathbf{a}_i}{\left\|{}^p\overrightarrow{A_iB_i}\right\|_2} = \frac{{}^p\mathbf{b}_i - {}^p\mathbf{R}_b {}^b\mathbf{a}_i - {}^p\mathbf{t}_b}{\left\|{}^p\overrightarrow{A_iB_i}\right\|_2} \tag{1.4}$$

Note that ${}^p\mathbf{R}_b$ and ${}^p\mathbf{t}_b$ are the rotation matrix and translation vector from frame $\mathcal{F}_p$ to frame $\mathcal{F}_b$.

Inverse Geometric Model with Pulleys

In reality, cable exit points are not fixed. Indeed, they move along the pulley sheave. To find out whether pulley geometry needs to be taken into account, the pulley radius and the average cable length need to be compared [Pot12]. The larger the pulley diameter, the more significant its influence on the moving-platform pose.

Pulleys of ACROBOT and CAROCA are shown in Figs. 1.13 and 1.14. In both cases, the pulleys have two DoF: the rotation axis of the pulley sheave and a vertical rotation axis. The latter allows the pulley to rotate as the moving-platform moves. Due to this, it is assumed that the cable does not twist, because it remains in a plane with its anchor point and pulley.

While ACROBOT's pulleys are small and the rotation axes intersect, this is not the case for CAROCA. Indeed, for the larger CDPR pulley diameter is considerably bigger. Furthermore, there is a distance between the vertical and pulley rotation axes that needs to be taken into account.

For pulleys of non-negligible size, the cable exit point has to be defined as the tangent point between the cable and the pulley sheave. The geometric model of a pulley is shown in Fig. 1.15. Here, the frame $\mathcal{F}_i$ of the ith pulley has its origin in point A_i and the axes are $\mathbf{x}_i$, $\mathbf{y}_i$ and $\mathbf{z}_i$. Axis $\mathbf{z}_i$ is vertical, $\mathbf{x}_i$ goes through A_i and the center of the pulley P_i, and

$y_i = z_i \times x_i$. The ith cable and the corresponding pulley sheave lie in the plane spanned by vectors x_i and z_i.

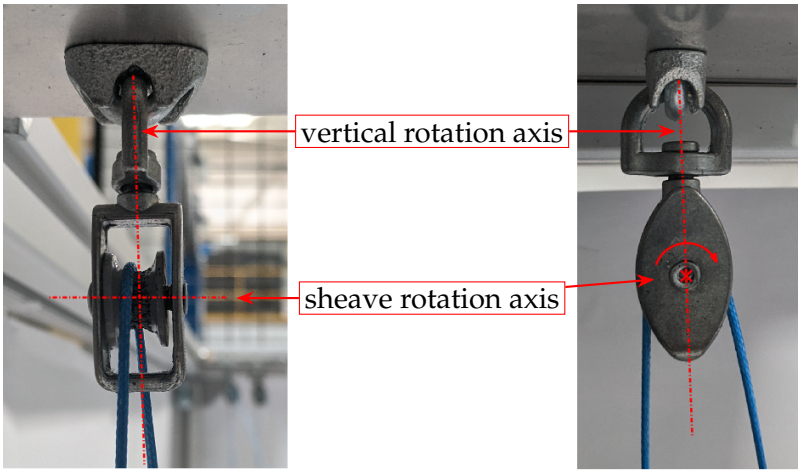


Figure 1.13: ACROBOT pulley sheave of diameter 9 mm

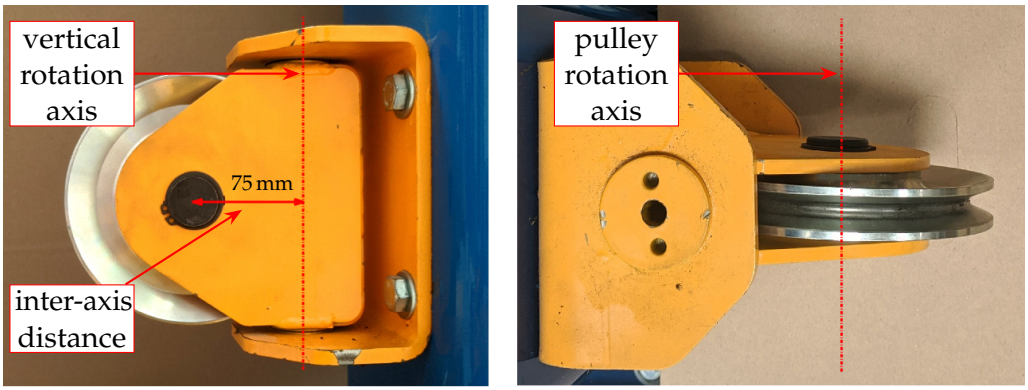


Figure 1.14: CAROCA pulley sheave of diameter 150 mm

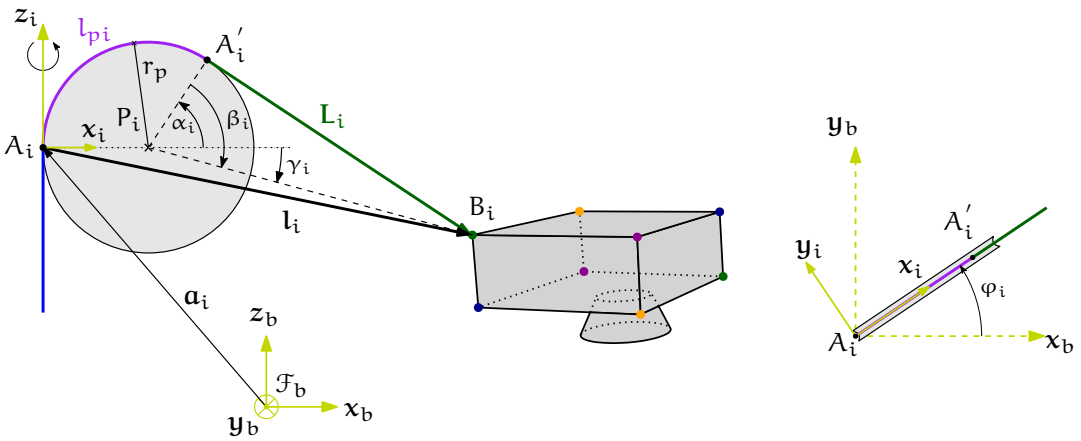


Figure 1.15: Pulley geometry

The actual ith cable exit point, that is, the tangent point to its pulley is noted as $A_i^{'}$ in Fig. 1.15. The vector pointing from P_i to $A_i^{'}$ is named $\mathbf{c}_i$ while the vector pointing from P_i to B_i is named $\mathbf{m}_i$. The exit point location on the pulley depends on the moving-platform pose. L_i is the length of the ith cable between points $A_i^{'}$ and B_i. l_{pi} denotes the length of the cable wrapped on the pulley between points A_i and $A_i^{'}$. The full cable length from A_i to B_i becomes:

$$l_{fi} = L_i + l_{pi} \tag{1.5}$$

with

$$l_{pi} = r_p(\pi - \alpha_i) \tag{1.6}$$

and the angle α_i is computed as:

$$\alpha_i = -\beta_i + \gamma_i \tag{1.7}$$

where:

$$\beta_i = -\text{atan2}(L_i,\, r_p) \tag{1.8}$$

and

$$\gamma_i = \arcsin\left(\frac{{}^b a_{iz} - {}^b b_{iz}}{\|\mathbf{m}_i\|_2}\right) \tag{1.9}$$

Note that here ${}^b a_{iz}$ and ${}^b b_{iz}$ are the third component of vectors ${}^b\mathbf{a}_i$ and ${}^b\mathbf{b}_i$, respectively.

Vector ${}^b\mathbf{c}_i$ can now be defined as:

$${}^b\mathbf{c}_i = \begin{bmatrix} r_p \cos(\varphi_i)\cos(\alpha_i) \\ r_p \sin(\varphi_i)\cos(\alpha_i) \\ r_p \sin(\alpha_i) \end{bmatrix} \tag{1.10}$$

The angle φ_i denotes the rotation of pulley plane about the axis $\mathbf{z}_i$. It can be computed as:

$$\varphi_i = \text{atan2}(l_{yi}, l_{xi}) \tag{1.11}$$

where l_{xi} and l_{yi} are the components of $\mathbf{l}_i$ along the $\mathbf{x}$ and $\mathbf{y}$ axes of the base frame $\mathcal{F}_b$. The vector ${}^b\mathbf{n}_i$ points from the origin of the base frame $\mathcal{F}_b$ to the pulley center-point P_i and is computed as:

$${}^b\mathbf{n}_i = {}^b\mathbf{a}_i + r_p\, {}^b\mathbf{R}_i\, \mathbf{x}_b = {}^b\mathbf{a}_i + r_p\, {}^b\mathbf{x}_i \tag{1.12}$$

with

$$ {}^{b}\mathbf{R}_i = \mathbf{R}_{zi}(\varphi_i) = \begin{bmatrix} \cos(\varphi_i) & -\sin(\varphi_i) & 0 \\ \sin(\varphi_i) & \cos(\varphi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.13} $$

and r_p is the radius of the pulley sheave.

Vector $\mathbf{m}_i$ is computed as:

$$ \mathbf{m}_i = {}^{b}\mathbf{b}_i - {}^{b}\mathbf{n}_i \tag{1.14} $$

The ith cable vector $\mathbf{L}_i$ is computed as:

$$ \mathbf{L}_i = \mathbf{m}_i - \mathbf{c}_i \tag{1.15} $$

Note that the length L_i can be computed simply by:

$$ L_i = \sqrt{\mathbf{m}_i\,\mathbf{m}_i^{\top} - r_p^2} \tag{1.16} $$

and the unit vector can be written as:

$$ {}^{p}\mathbf{U}_i = \frac{\mathbf{L}_i}{L_i} \tag{1.17} $$

1.3.2 Kinematic Modeling

The cable velocity vector $\dot{\mathbf{l}}$ is obtained upon differentiation of (1.2) with respect to time:

$$ \dot{\mathbf{l}} = \mathbf{A}\,{}^{p}\mathbf{v}_p \tag{1.18} $$

where ${}^{p}\mathbf{v}_p$ is the moving-platform twist expressed in its own frame $\mathcal{F}_p$ and $\mathbf{A}$ is the Forward Jacobian matrix of the CDPR. For a CDPR with negligible pulleys it is defined as [Pot18]:

$$ \mathbf{A} = \begin{bmatrix} {}^{p}\mathbf{u}_1^{\top} & ({}^{p}\mathbf{b}_1 \times {}^{p}\mathbf{u}_1)^{\top} \\ \vdots & \vdots \\ {}^{p}\mathbf{u}_m^{\top} & ({}^{p}\mathbf{b}_m \times {}^{p}\mathbf{u}_m)^{\top} \end{bmatrix} \tag{1.19} $$

where $m = 8$ for a spatial CDPR with eight cables, which makes the Jacobian $\mathbf{A}$ a (8×6)–matrix.

Note that in case pulley geometry needs to be taken into account, then the Forward Jacobian becomes:

$$\mathbf{A_{II}} = \begin{bmatrix} {}^{p}\mathbf{U}_1^{\top} & ({}^{p}\mathbf{b}_1 \times {}^{p}\mathbf{U}_1)^{\top} \\ \vdots & \vdots \\ {}^{p}\mathbf{U}_m^{\top} & ({}^{p}\mathbf{b}_m \times {}^{p}\mathbf{U}_m)^{\top} \end{bmatrix} \tag{1.20}$$

The instantaneous inverse kinematic model depends on constant parameters ${}^{b}\mathbf{a}_i$ and ${}^{p}\mathbf{b}_i$, as well as the homogeneous transformation ${}^{p}\mathbf{T}_b$ between the moving-platform frame $\mathcal{F}_p$ and the base frame $\mathcal{F}_b$. Thus, the knowledge of the current moving-platform pose in $\mathcal{F}_b$ is necessary.

Finally, the motor velocities $\dot{\mathbf{q}}_m$ are directly proportional to cable velocities:

$$\dot{\mathbf{q}}_m = \frac{1}{r_w}\dot{\mathbf{l}} \tag{1.21}$$

where r_w is the drum radius of the winches mounted on the shaft of each motor.

It is also possible to express the moving-platform twist in the base frame $\mathcal{F}_b$. The expression of the corresponding Jacobian is detailed in Appendix A.3.1.

The Kinematic Model of a Planar CDPR

For stability analysis, described in Section 3.2.1, a planar CDPR is used as a simplification to be able to write analytical expressions. For this reason, in this section we present the kinematic equations of a planar CDPR, the schematic of which is shown in Fig. 1.16.

This planar CDPR is actuated by four cables to achieve three-DoF and perform a 2T1R motion type. To keep as much similarity with the spatial CDPRs as possible, here too, the camera is mounted on the moving-platform, thus the transformation ${}^{p}\mathbf{T}_c$ between the frames $\mathcal{F}_p$ and $\mathcal{F}_c$ does not change with time, unlike ${}^{b}\mathbf{T}_p$ between $\mathcal{F}_b$ and $\mathcal{F}_p$, and ${}^{c}\mathbf{T}_o$ between $\mathcal{F}_c$ and $\mathcal{F}_o$.

In general, the cable velocity equation (1.18) obtained for the spatial CDPR applies also for the planar CDPR. However, it must be noted that the dimensions of all variables are reduced: (i) rotation matrices ${}^{i}\mathbf{R}_j$ are (2×2)–matrices; (ii) homogeneous transformation matrices ${}^{i}\mathbf{T}_j$ are (3×3)–matrices; (iii) velocity vectors ${}^{i}\mathbf{v}_j$ are of size (3×1); (iv) and coordinates ${}^{p}\mathbf{b}_i$ and ${}^{b}\mathbf{a}_i$, as well as translation vectors ${}^{i}\mathbf{t}_j$ are of size (2×1).

Thus, for the planar CDPR (1.18) becomes:

$$\dot{\mathbf{l}}_{pl} = \mathbf{A}_{pl}\,{}^{p}\mathbf{v}_p \tag{1.22}$$

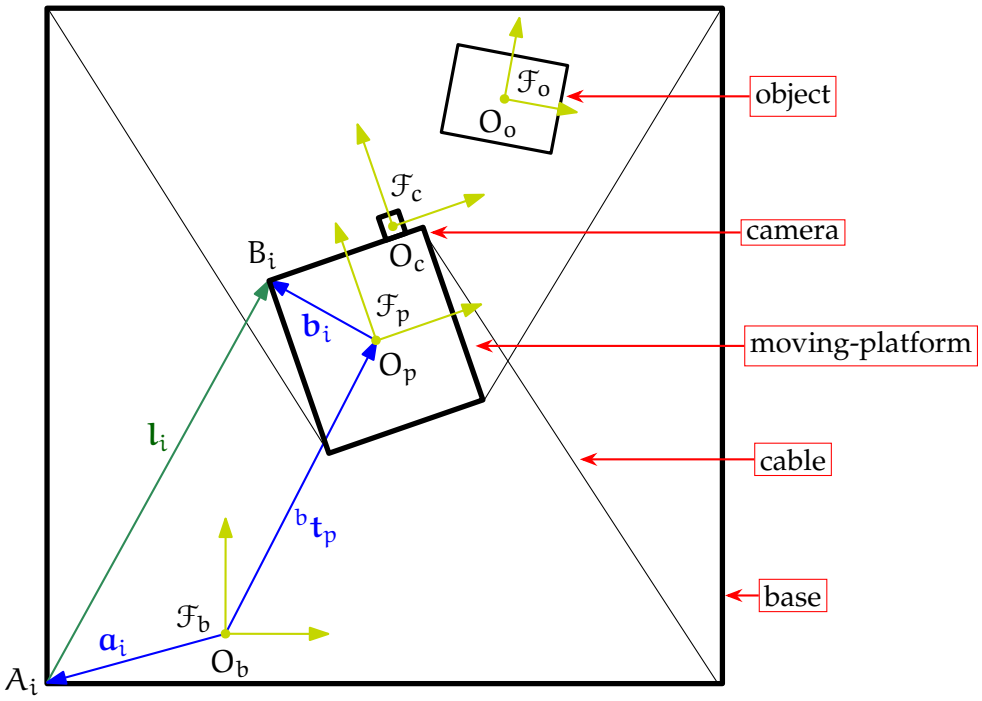


Figure 1.16: Schematics of a planar CDPR with 4 cables, a camera mounted on its moving-platform and an object in the workspace

where in the planar case $\dot{\mathbf{l}}_{pl}$ is a (4×1)–vector, and $\mathbf{A}_{pl}$ is the Forward Jacobian, which takes the following form [GA88]:

$$\mathbf{A}_{pl} = \begin{bmatrix} {}^p\mathbf{u}_1^\top & {}^p\mathbf{b}_1^\top \mathbf{E}^\top {}^p\mathbf{u}_1 \\ \vdots & \vdots \\ {}^p\mathbf{u}_m^\top & {}^p\mathbf{b}_m^\top \mathbf{E}^\top {}^p\mathbf{u}_m \end{bmatrix} \tag{1.23}$$

where $\mathbf{E} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$.

1.3.3 Static Feasible Workspace

The static equilibrium of the moving-platform is defined as:

$$\boldsymbol{W}\boldsymbol{\tau} + \boldsymbol{w}_g = \mathbf{0}_6 \tag{1.24}$$

where $\boldsymbol{W}$ is the wrench matrix and it is related to the Jacobian as $\boldsymbol{W} = -\boldsymbol{A}^\top$; $\boldsymbol{\tau}$ is the vector of cable tensions and $\boldsymbol{w}_g$ is the gravity wrench.

Based on the static equilibrium equation (1.24), it is possible to define the Static-Feasible Workspace (SFW) as [Gag+16]:

$$\mathcal{S} = \{\mathbf{p}_p \in SE(3) : \exists \boldsymbol{\tau} \in \mathcal{T},\ \boldsymbol{W}\boldsymbol{\tau} + \boldsymbol{w}_g = \mathbf{0}_6\} \tag{1.25}$$

Namely, the workspace $\mathcal{S}$ is the set of all moving-platform poses $\mathbf{p}_p$ for which there exists a vector of cable tensions $\boldsymbol{\tau}$ within the cable tension space $\mathcal{T}$ such that the CDPR can balance the gravity wrench $\boldsymbol{w}_g$, and $\boldsymbol{W}\boldsymbol{\tau} + \boldsymbol{w}_g = \mathbf{0}_6$. The tension space $\mathcal{T}$ is an m-dimensional box of feasible tensions [Rui+15]:

$$\mathcal{T} = \{\boldsymbol{\tau} \in \mathbb{R}^m \,:\, \boldsymbol{\tau}_{lb} \leqslant \boldsymbol{\tau} \leqslant \boldsymbol{\tau}_{ub}\} \tag{1.26}$$

where $\boldsymbol{\tau}_{lb}$ and $\boldsymbol{\tau}_{ub}$ are the lower and upper tension bounds, respectively.

Static Feasible Workspace of ACROBOT

The Static Workspace of ACROBOT was traced using the ARACHNIS software for the analysis and parametric design of CDPRs [Rui+15]. The shape of the workspace depends on multiple input parameters, including tension bounds and moving-platform shape and

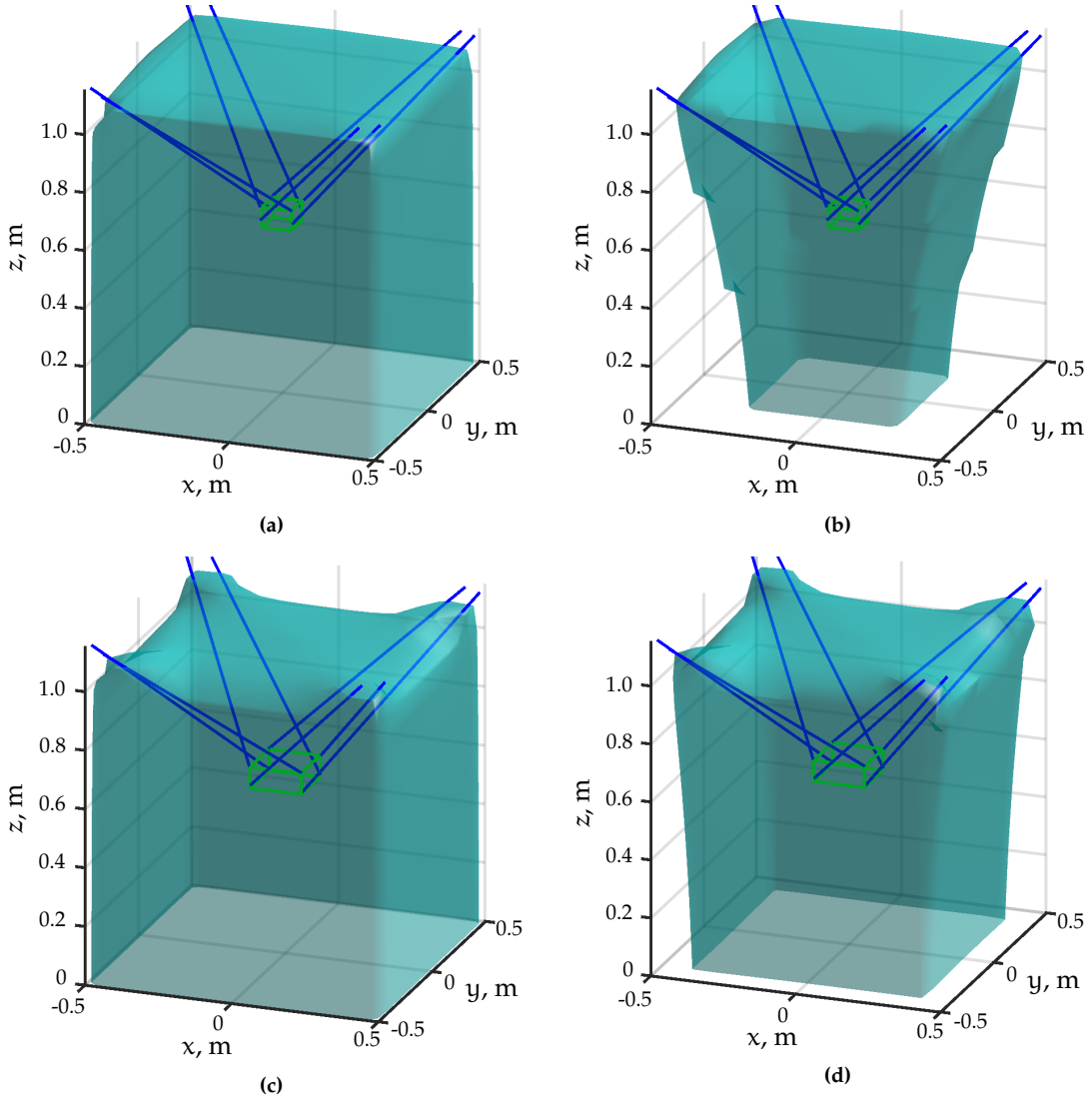


Figure 1.17: SFW for ACROBOT: **(a)** small moving-platform and $\tau_{lb} = 0\,\text{N}$; **(b)** small moving-platform and $\tau_{lb} = 1\,\text{N}$; **(c)** big moving-platform and $\tau_{lb} = 0\,\text{N}$; **(d)** big moving-platform and $\tau_{lb} = 1\,\text{N}$

weight. Four examples are shown in Fig. 1.17. Both of the moving-platform sizes were tested with lower bound tension τ_{lb} being equal to 0 N or 1 N.

Static Feasible Workspace of CAROCA

The Static Workspace of CAROCA was also traced using the ARACHNIS software [Rui+15]. Only one moving-platform is used in this thesis, however the shape of the workspace depends on tension bounds. Two examples are shown in Fig. 1.18 with either $\tau_{lb} = 0\,\text{N}$ or $\tau_{lb} = 30\,\text{N}$.

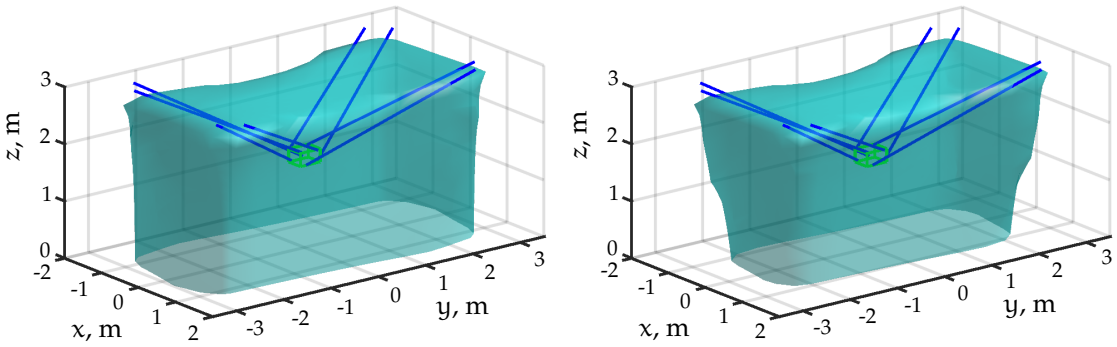


Figure 1.18: SFW for CAROCA: **(a)** $\tau_{lb} = 0\,\text{N}$; **(b)** $\tau_{lb} = 30\,\text{N}$

1.3.4 Tension Distribution Algorithm

Introduction

It is possible to determine if the current moving-platform pose is in equilibrium with given cable tensions according to (1.24). Unfortunately, it is tricky to find the tensions that would keep the moving-platform in the static equilibrium. This is because using simply $\boldsymbol{\tau} = -\boldsymbol{W}^{\dagger}\boldsymbol{w}_g$, can lead to negative tensions in one or more cables, which means that the cable would need to push the moving-platform. While this would not be a problem for a Gough-Stewart platform, the cables of a CDPR can only pull, but they cannot push. Instead the effect of a negative tension will be the creation of a sag in the cable, which is undesirable.

For this reason, a Tension Distribution Algorithm (TDA) is used usually instead. It computes the cable tensions so that the overall effort on the moving-platform is divided among all the cables.

Using TDA is particularly important for fully constrained CDPRs, because its cables are pulling in all directions and can create very high tensions that will finally lead to the CDPR breaking. On the other hand, for suspended CDPRs it is used less often. This is because without cables pulling from the bottom, it is unlikely to attain the tensions computed by a TDA, unless the CDPR has very elastic cables. However, a TDA could be used to determine whether the cables are slack or not, by comparing the TDA output and cable tension measurements.

There exist many tension distribution algorithms. The most notable are the Median method [Pot14], which has a very low computation cost, and the Barycenter method [Mik+08] [Gou+15].

A CDPR with eight cables and six–DoF has a redundancy degree $r = m - n = 8 - 6 = 2$. For such CDPRs a TDA based on the computation of a feasible tension polygon is used [HK11]. Gouttefarde *et al.* proposed a real-time capable feasible polygon computation for the Barycenter method in [Gou+15]. This method has a limited amount of time to find a solution, but if one is found it will respect all the given tension constraints.

A particular solution for the static equilibrium defined in (1.24) can be written as:

$$\boldsymbol{\tau} = \boldsymbol{W}^{\dagger}\boldsymbol{w}_g + \mathbf{N}\boldsymbol{\eta} \tag{1.27}$$

where $\mathbf{N}$ is the $(m \times 2)$ null space matrix of the wrench matrix $\boldsymbol{W}$ and $\boldsymbol{\eta} = \begin{bmatrix}\eta_1 & \eta_2\end{bmatrix}^{\top}$ is an arbitrary vector. The term $\mathbf{N}\boldsymbol{\eta}$ moves the particular solution $\boldsymbol{\tau}_p = \boldsymbol{W}^{\dagger}\boldsymbol{w}_g$ into the feasible range of cable tensions. The mentioned feasible range is defined by a lower bound τ_{lb} and upper bound τ_{ub}.

The Barycenter Method

There exists an intersection Λ between the 2D space of solution obtained from (1.27), defined as Σ, with the hypercube of feasible cable tension $\Omega = \{\tau \mid 0 \leqslant \tau_{lb} \leqslant \tau_i \leqslant \tau_{ub}\}$:

$$\Lambda = \Sigma \cap \Omega \tag{1.28}$$

Λ is a 2D convex polytope of feasible cable tension sets, also known as feasible polygon. Such a polygon exists if and only if there is at least one solution that satisfies all cable tension limits as well as the static equilibrium of the moving-platform (1.24). The feasible polygon is defined in the $\boldsymbol{\eta}$–space by the following linear inequalities:

$$\boldsymbol{\tau}_{lb} - \boldsymbol{\tau}_p \leqslant \mathbf{N}\boldsymbol{\eta} \leqslant \boldsymbol{\tau}_{ub} - \boldsymbol{\tau}_p \tag{1.29}$$

The feasible polygon, such as in Fig. 1.19, shows all the possible solutions. It is then necessary to choose one of the solutions. In the Barycenter method, as the name suggests, the barycenter of the feasible polygon is chosen (red dot in the middle of the polygons shown in Fig. 1.19). This method chooses the solution that is as far as possible from all of the tension limits.

As an example, the feasible polygon is computed for ACROBOT with the large moving-platform based on the SFW shown in Fig. 1.17c. The feasible polygon shown in Fig. 1.19a corresponds to the moving-platform being at the center of the workspace, so that its pose is ${}^{b}\mathbf{p}_p = \begin{bmatrix}0\,\text{m} & 0\,\text{m} & 0.15\,\text{m} & 0^\circ & 0^\circ & 0^\circ\end{bmatrix}^{\top}$. At this pose all the cables should be equally tensed. Indeed, the tension set found with the Barycenter method is $\boldsymbol{\tau} = \begin{bmatrix}5.22\,\text{N} & 5.46\,\text{N} & 5.16\,\text{N} & 5.41\,\text{N} & 5.13\,\text{N} & 5.39\,\text{N} & 5.22\,\text{N} & 5.46\,\text{N}\end{bmatrix}$.

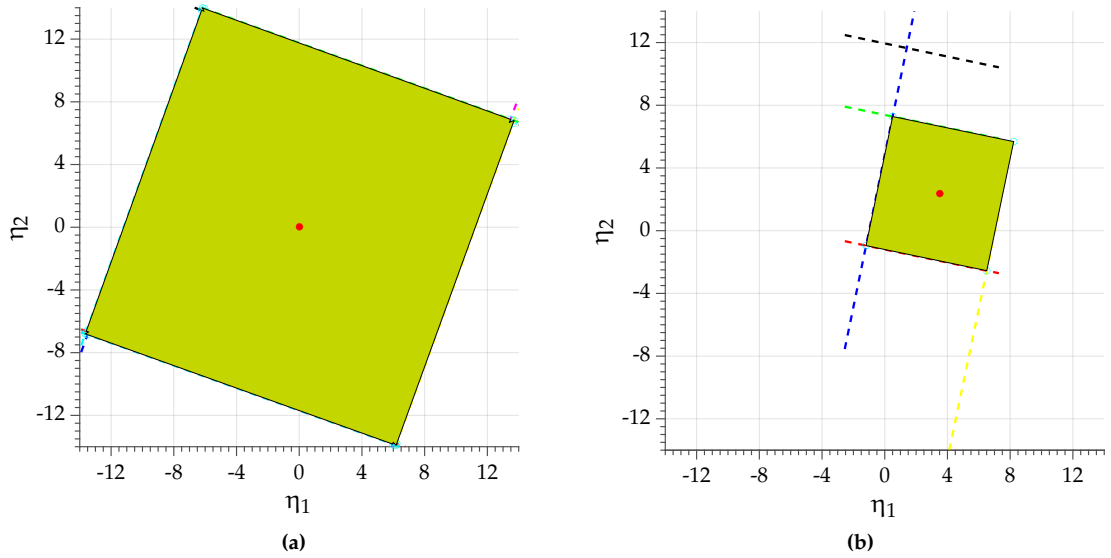


Figure 1.19: Feasible polygon plots: **(a)** moving-platform in homing pose; **(b)** moving-platform close to the boundary of the workspace

The feasible polygon shown in Fig. 1.19b corresponds to the moving-platform pose being ${}^{b}\mathbf{p}_{p} = \begin{bmatrix} 0.35\,m & -0.27\,m & 0.45\,m & 0° & 0° & 0° \end{bmatrix}^{\top}$. As a result, the tension set is now $\boldsymbol{\tau} = \begin{bmatrix} 2.08\,N & 2.08\,N & 2.43\,N & 2.58\,N & 3.74\,N & 4.30\,N & 12.76\,N & 13.31\,N \end{bmatrix}$. The change in moving-platform position causes cable $\mathcal{C}_7$ and $\mathcal{C}_8$ to be significantly more tensed than others. Furthermore, the size of the feasible polygon has significantly decreased.

1.4 Visual Servoing

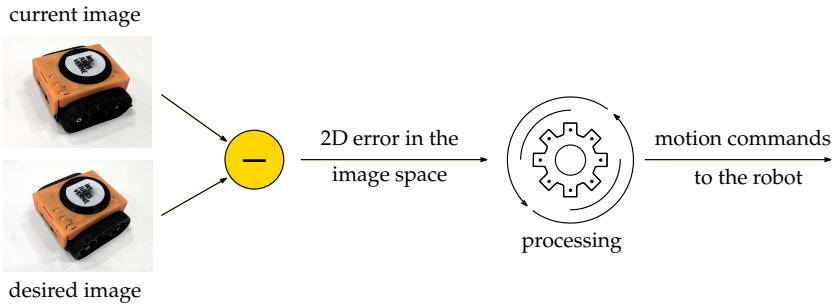
1.4.1 Introduction

Visual servoing, also known as vision-based robot control and abbreviated as VS, is a technique that uses vision sensors (cameras) in the feedback loop of the control algorithm to control the motion of a robot. The goal is to reduce the error $\boldsymbol{e} = \mathbf{s}(\mathbf{f}(t), \boldsymbol{\alpha}) - \mathbf{s}^{*}$, where the content of the feature vector $\mathbf{s}(\mathbf{f}(t), \boldsymbol{\alpha})$ and its desired value $\mathbf{s}^{*}$ depend on the control technique [CH08]:

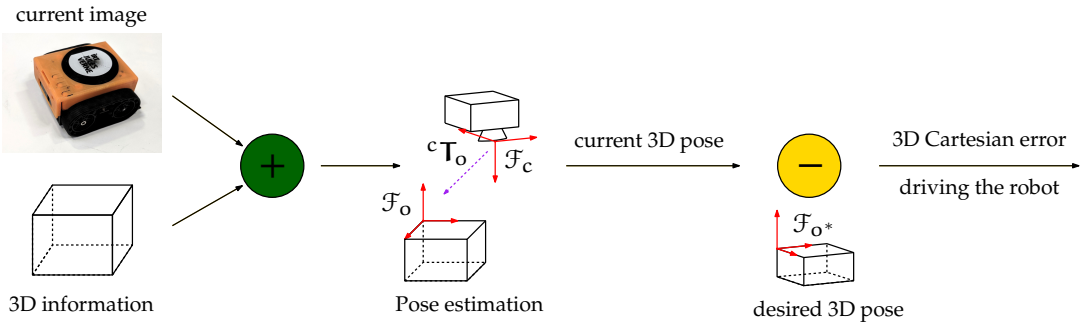
- Image-based VS (IBVS), where the control is done in the image plane, as can be seen in Fig. 1.20a. This approach does not necessitate an estimation of a Cartesian pose, instead $\mathbf{s}$ consists of image-based features $\mathbf{f}(t)$, such as corners or image moments; and the constant calibration parameters $\boldsymbol{\alpha}$. The controller is not well suited for large-scale motion, because it can get stuck in a local minimum.
- Pose-based VS (PBVS), where the control is done in the Euclidean space, as can be seen in Fig. 1.20b. In this approach, the visual primitive $\mathbf{s}$ is a pose, which is computed from some image features $\mathbf{f}(t)$, the constant calibration parameters and the 3D model of the object in $\boldsymbol{\alpha}$. The estimated pose is compared to the desired pose, and the difference allows to determine the necessary motion. Due to the composition

of $\mathbf{s}$, errors in the pose estimation leave a significant impact on the accuracy and stability of the control.

- Hybrid approaches use some combination of 2D (image-based) and 3D (pose-based) servoing. The most popular examples are [CH08]:
  - 2½D Visual Servoing, that takes advantage of IBVS and PBVS benefits, while not suffering from their drawbacks. Here the rotational and translational motions are decoupled. The feature vector $\mathbf{s}$ contains both 2D and 3D features, due to which the controller does not get stuck in local minima and the impact of errors in pose estimation on accuracy is reduced.
  - Partitioned approach, where the motions related to $\mathbf{x}$ and $\mathbf{y}$ axes is decoupled from those to $\mathbf{z}$ axis. Unlike 2½D VS, all of the features are expressed directly in the image. The visual servoing task is split in two: the first task is to keep the features in the field of view; the second task is to bring the camera to the desired pose.
  - Switching approach, where both controllers (image-based and pose-based) are used sequentially, and the switches are determined based on some Lyapunov functions for these controllers.



(a) Image-based Visual Servoing



(b) Pose-based Visual Servoing

Figure 1.20: Image-based and Pose-based visual servoing visualization

Furthermore, as can be seen in Figs. 1.21a and 1.21b, regarding the position of the camera, there can be two configurations:

- Eye-in-hand, where the camera is mounted on (or close to) the end-effector of the robot. The camera is moving together with the end-effector.

- Eye-to-hand, where the camera is fixed in the environment so that the end-effector of the robot is within the field of view. Generally, the camera stays in the same location while the end-effector moves. However, it is also possible that the camera is actuated separately from the robot.

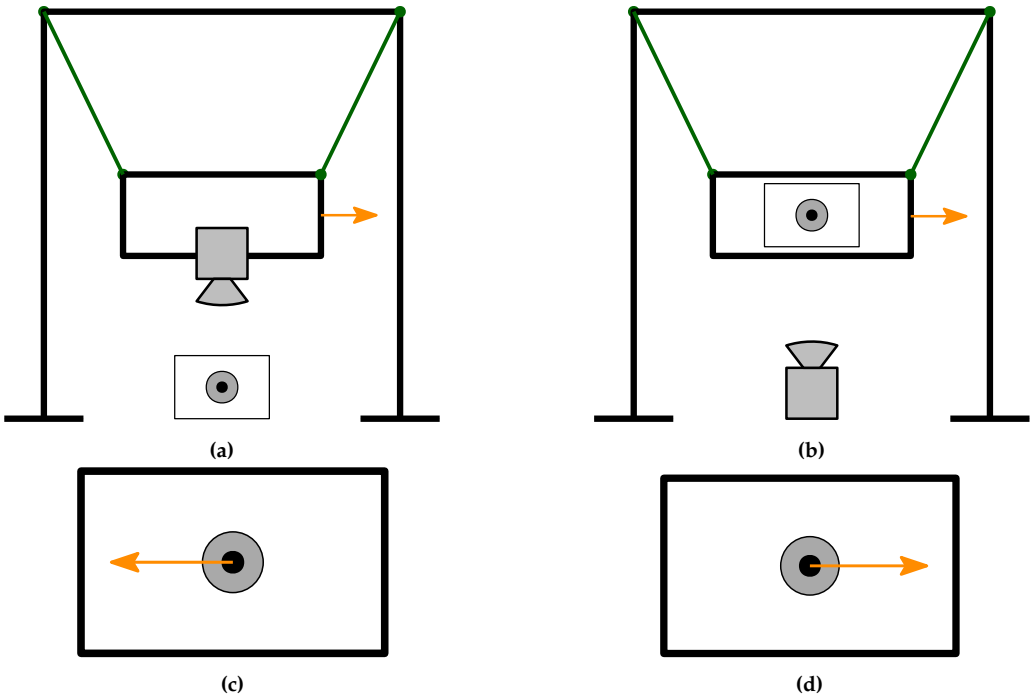


Figure 1.21: Camera configurations: **(a)** eye-in-hand; **(b)** eye-to-hand. Corresponding perception in the image of the same robot motion: **(c)** eye-in-hand; **(d)** eye-to-hand.

When deciding which configuration to use, usually one should consider the task at hand. The main advantages and drawbacks are summed up in Table 1.1. Note that the choice of camera configuration also affects the hand-eye calibration problem. Indeed in eye-in-hand configuration the hand-eye calibration consists of recovering the camera pose with respect to the end-effector. On the contrary, in eye-to-hand configuration the hand-eye calibration is used to obtain the camera pose in the base frame. The same robot motion is perceived differently depending on the chosen camera configuration, as can be seen in Fig. 1.21.

The interest of using visual servoing is in its robustness with respect to modeling and calibration errors, as well as robustness to slight changes in the environment. If there is a visual feedback, then the object does not have to be placed in exactly the same position to be picked up correctly by the robot. Indeed, robot motion is adjusted as a function of object location as seen by the camera. Furthermore, the target can be tracked during its motion.

Of course the drawback of such an approach is in the load on the processor: many images need to be processed in a short period of time to ensure system stability and high accuracy of the motion.

Table 1.1: Camera configuration differences

Eye-in-hand	Eye-to-hand
The field of view changes with robot motion	The field of view is constant
The end-effector (or the moving-platform in CDPRs) cannot occlude the object of interest	The end-effector can occlude the object by approaching it
Since the camera moves with the end-effector, it is possible to locate the object of interest by doing a control sweep over the workspace, thus if the object is in the workspace it will be found	The object can be out of the field of view, usually multiple cameras are necessary to cover the whole workspace
Can achieve high accuracy with respect to the object, because the camera approaches the object with the end-effector motion and the closer it is the more precise the computer vision algorithm can be.	The further the object from the camera, the lower the accuracy. Using multiple cameras can improve accuracy but still will not be as good as for eye-in-hand
The end-effector is not observed. Thus, for a CDPR the moving-platform pose must be estimated. The estimation is often open-loop, which can lead to estimation error accumulation and to loss of stability.	The moving-platform is observed by the camera, thus its pose is known in the camera frame.
In pick&place tasks or similar special care is necessary to avoid covering the field of view when the object is picked to be able to perceive the place location.	Picking the object does not cover the camera, no problem in guiding the robot to the place location.

1.4.2 The Principle of Visual Servoing

No matter the chosen visual servoing approach, the main principles remain the same and they are introduced in this section. As mentioned before, the task of visual servoing is to reduce the error $\boldsymbol{e}$ between the current feature vector $\mathbf{s}$ and its desired value $\mathbf{s}^*$. To do so, the most basic strategy is to select an exponential decoupled decrease of the error:

$$\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e} \tag{1.30}$$

where λ is a positive gain that can be either constant or adaptive. In the latter case, λ can be designed as [ViS19]:

$$\lambda(x) = (\lambda_0 - \lambda_\infty) e^{-(\dot{\lambda}_0/(\lambda_0-\lambda_\infty))x} + \lambda_\infty \tag{1.31}$$

where:

- $x = \|\boldsymbol{e}\|_2$ is the 2–norm of error $\boldsymbol{e}$ at the current iteration
- $\lambda_0 = \lambda(0)$ is the gain tuned for very small values of x

- $\lambda_\infty = \lambda(\infty)$ is the gain tuned for very high values of x
- $\dot{\lambda}_0$ is the slope of λ at $x = 0$

Upon derivation of $\boldsymbol{e} = \mathbf{s} - \mathbf{s}^*$ with respect to time, we get:

$$\dot{\boldsymbol{e}} = \mathbf{L_s}\,{}^{c}\mathbf{v_c} \tag{1.32}$$

where $\mathbf{L_s}$ is the interaction matrix, also known as the Feature Jacobian, and its composition depends on the chosen visual features. Furthermore, ${}^{c}\mathbf{v_c}$ is the Cartesian velocity of the camera, expressed in its own frame $\mathcal{F}_c$.

It is then possible to express the camera velocity from (1.30) and (1.32):

$${}^{c}\mathbf{v_c} = -\lambda\,\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\boldsymbol{e} \tag{1.33}$$

where $\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}$ is the pseudo-inverse of the interaction matrix estimation.

The closed-loop equation is expressed by injecting (1.33) into (1.32):

$$\dot{\boldsymbol{e}} = -\lambda\,\mathbf{L_s}\,\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\boldsymbol{e} \tag{1.34}$$

According to Lyapunov stability analysis, the system remains globally asymptotically stable as long as its stability criterion $\mathbf{\Pi} = \mathbf{L_s}\,\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}$ is positive definite [Kha02]. Indeed, it is clear from (1.34) that as long as $\mathbf{\Pi} > \mathbf{0}$, the error $\boldsymbol{e}$ converges to zero.

Note that the camera velocity ${}^{c}\mathbf{v_c}$ is proportional to the error $\boldsymbol{e}$. When using a constant gain λ the velocity is very large at the beginning and becomes small as the desired vector $\mathbf{s}^*$ is approached. This leads to a sudden jerk in the beginning of the trajectory and a very low velocity at the end of trajectory. The velocity can be so low that some of the actuators do not overcome the static friction in their transmission and the overall motion becomes asynchronous [RCM14]. Because of this, the adaptive gain, defined in (1.31) should be preferred. Furthermore, a continuous velocity should be implemented to deal with the jerk in the beginning of the trajectory. The jerk occurs because the initial velocity is $\mathbf{0}$ and as soon as it changes the acceleration becomes infinitely large for a short instant. A continuous gain is applied to velocity ${}^{c}\mathbf{v_c}$ as follows [MC07]:

$${}^{c}\mathbf{v_c} = -\lambda\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\boldsymbol{e} + \lambda\widehat{\mathbf{L}}_{\mathbf{s}_{\boldsymbol{e}(0)}}^{\dagger}\boldsymbol{e}(0)\mathrm{e}^{(-\sigma t)} \tag{1.35}$$

where the first part corresponds to (1.33), while the second part ensures the continuity of velocities. Furthermore, t is the time spent since the beginning of the task; $\boldsymbol{e}(0)$ is the initial value of $\boldsymbol{e}$ at time $t = 0\,\mathrm{s}$; $\widehat{\mathbf{L}}_{\mathbf{s}_{\boldsymbol{e}(0)}}^{\dagger}$ is the pseudo-inverse of the interaction matrix, computed for $\boldsymbol{e} = \boldsymbol{e}(0)$; and σ is a constant gain that needs to be tuned. This allows us to have a smooth transition from initial velocity of $\mathbf{0}$ to the computed velocity, necessary to fulfill the task.

The Principle of Visual Servoing of CDPRs

A generic control scheme for visual servoing of a CDPR is shown in Fig. 1.22.

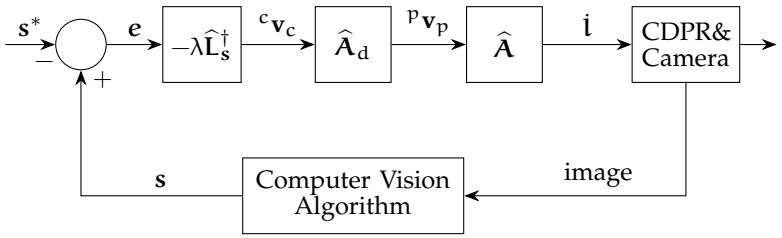


Figure 1.22: Control scheme for visual servoing of a CDPR

The camera is mounted on the moving-platform in the eye-in-hand configuration. Comparing the current feature vector $\mathbf{s}$, which is available from a computer vision algorithm, to the desired feature vector $\mathbf{s}^*$, which is previously defined, it is possible to compute the desired cable velocities that will make $\mathbf{s}$ converge to $\mathbf{s}^*$.

The relation between the camera velocity ${}^{c}\mathbf{v}_c$ and the moving-platform velocity ${}^{p}\mathbf{v}_p$ is expressed as:

$$ {}^{p}\mathbf{v}_p = \mathbf{A}_d \, {}^{c}\mathbf{v}_c \tag{1.36} $$

where $\mathbf{A}_d$ is the Adjoint matrix and in this case it takes the following form [KD04] [Sic+10]:

$$ \mathbf{A}_d = \begin{bmatrix} {}^{p}\mathbf{R}_c & [{}^{p}\mathbf{t}_c]_\times \, {}^{p}\mathbf{R}_c \\ \mathbf{0}_3 & {}^{p}\mathbf{R}_c \end{bmatrix} \tag{1.37} $$

where ${}^{p}\mathbf{t}_c$ and ${}^{p}\mathbf{R}_c$ are the translation vector and rotation matrix that form the homogeneous transformation matrix ${}^{p}\mathbf{T}_c$ from $\mathcal{F}_p$ to $\mathcal{F}_c$.

The model of the system shown in Fig. 1.22 is written from Eqs. (1.18), (1.32) and (1.36):

$$ \dot{\boldsymbol{e}} = \mathbf{L}_s \, \mathbf{A}_d^{-1} \mathbf{A}^\dagger \, \dot{\mathbf{l}} \tag{1.38} $$

where $\mathbf{A}^\dagger$ is the Moore-Penrose pseudo-inverse of the Jacobian matrix $\mathbf{A}$.

Upon injecting (1.36) and (1.33) into (1.18), the output of the control scheme, i.e. the cable velocity vector $\dot{\mathbf{l}}$, takes the form:

$$ \dot{\mathbf{l}} = -\lambda \, \widehat{\mathbf{A}} \, \widehat{\mathbf{A}}_d \, \widehat{\mathbf{L}}_s^\dagger \, \boldsymbol{e} \tag{1.39} $$

where $\widehat{\mathbf{A}}$ and $\widehat{\mathbf{A}}_d$ are the estimations of $\mathbf{A}$ and $\mathbf{A}_d$, respectively.

Finally, the closed-loop equation is expressed by injecting (1.39) into (1.38):

$$ \dot{\boldsymbol{e}} = -\lambda \, \mathbf{L}_s \, \mathbf{A}_d^{-1} \mathbf{A}^\dagger \, \widehat{\mathbf{A}} \, \widehat{\mathbf{A}}_d \, \widehat{\mathbf{L}}_s^\dagger \, \boldsymbol{e} \tag{1.40} $$

Note that the control can also be expressed in the base frame $\mathcal{F}_b$. This is done in Appendix A.3.

1.4.3 Image-Based Visual Servoing

As the name suggests, for IBVS the feature vector $\mathbf{s}$ consists of 2D image features. More precisely, $\mathbf{s}$ usually consists of normalized point coordinates, however it is not the only choice [CH08].

For a point $\mathbf{K}$ the Cartesian coordinates (X, Y, Z) expressed in the camera frame project into the image plane as a point $\mathbf{k} = (x, y)$ in the following way:

$$\begin{cases} x = \dfrac{X}{Z} = \dfrac{u - u_0}{p_x} \\ y = \dfrac{Y}{Z} = \dfrac{v - v_0}{p_y} \end{cases} \tag{1.41}$$

where (u, v) are the pixel coordinates of the point; (u_0, v_0) are the pixel coordinates of image center; $p_x = f/l_x$ and $p_y = f/l_y$ with f being the focal length and l_x and l_y being the pixel size along $\mathbf{x}$ and $\mathbf{y}$ axes, respectively.

Then the velocity of point $\mathbf{k}$ is related to the camera velocity as:

$$\dot{\mathbf{k}} = \mathbf{L}_k \, {}^c\mathbf{v}_c \tag{1.42}$$

where the interaction matrix $\mathbf{L}_k$ is expressed as:

$$\mathbf{L}_k = \begin{bmatrix} -\dfrac{1}{Z} & 0 & \dfrac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\dfrac{1}{Z} & \dfrac{y}{Z} & (1+y^2) & -xy & -x \end{bmatrix} \tag{1.43}$$

As can be seen in (1.43), the matrix $\mathbf{L}_k$ contains the Z coordinate of the 3D point $\mathbf{K}$. Thus, it is necessary to approximate this distance during the visual servo.

To be able to control six degrees of freedom, at least three points are necessary. In such a case the feature vector becomes $\mathbf{s} = (\mathbf{k}_1, \mathbf{k}_2, \mathbf{k}_3)$ and similarly the final interaction matrix $\mathbf{L}_\mathbf{s}$ is a stack of interaction matrices $\mathbf{L}_k$:

$$\mathbf{L}_\mathbf{s} = \begin{bmatrix} \mathbf{L}_{k_1} \\ \mathbf{L}_{k_2} \\ \mathbf{L}_{k_3} \end{bmatrix} \tag{1.44}$$

However, it is known that with just three points in some configurations $\mathbf{L}_\mathbf{s}$ is singular. Furthermore, there exist four different camera poses for which $\boldsymbol{e} = \mathbf{0}$. That is, there are four global minima and it is not possible to differentiate between them [CH08]. For this reason usually the feature vector $\mathbf{s}$ consists of at least four points.

It should be noted that having more than three points in the feature vector affects system stability. Indeed, for a system to be globally asymptotically stable, we need $\mathbf{L_s}\,\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger} > \mathbf{0}$. Using 3 points we get $j = 6$ features, which coincides with the amount of degrees of freedom that the camera can have. Then for 4 points we get $j = 8$ features, meaning that the multiplication $\mathbf{L_s}\,\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}$ is a 8×8-matrix, which is at most of rank 6. Thus, $\mathbf{L_s}\,\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}$ has a nontrivial null space and that results in having local minima in configurations that correspond to error $\boldsymbol{e}$ belonging to the kernel of $\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}$. Therefore, only local asymptotic stability can be obtained for IBVS [CH08].

1.4.4 Pose-Based Visual Servoing

In case of PBVS, the feature vector $\mathbf{s}$ consists of 3D features. It is assumed that once the object is seen in the image, it is possible to retrieve its Cartesian pose and compute the feature vector $\mathbf{s}$. It is then compared to the desired feature vector $\mathbf{s}^*$ and an error $\boldsymbol{e} = \begin{bmatrix}\boldsymbol{e}_t^\top & \boldsymbol{e}_\omega^\top\end{bmatrix}^\top$ is computed. Here, $\boldsymbol{e}_t$ is the translational error and it may be selected as $\boldsymbol{e}_t = {}^c\mathbf{t}_o - {}^c\mathbf{t}_o^*$ with ${}^c\mathbf{t}_o$ being the translational vector between the object frame $\mathcal{F}_o$ and the camera frame $\mathcal{F}_c$. Furthermore, $\boldsymbol{e}_\omega$ is the rotational error that is defined as $\boldsymbol{e}_\omega = \theta\mathbf{u}$, where θ is the angle and $\mathbf{u}$ is the axis of the rotational matrix ${}^{c^*}\mathbf{R}_c = {}^{c^*}\mathbf{R}_{o^*}\,{}^c\mathbf{R}_o^\top = {}^{c^*}\mathbf{R}_o\,{}^o\mathbf{R}_c$.

Upon derivation of $\boldsymbol{e}$ with respect to time, we get:

$$\dot{\boldsymbol{e}} = \begin{cases} \dot{\boldsymbol{e}}_t = \begin{bmatrix}-\mathbf{I}_3 & [\boldsymbol{e}_t]_\times\end{bmatrix} {}^c\mathbf{v}_c \\ \dot{\boldsymbol{e}}_\omega = \frac{d(\theta\mathbf{u})}{dt} = \begin{bmatrix}\mathbf{0}_3 & \mathbf{L}_\omega\end{bmatrix} {}^c\mathbf{v}_c \end{cases} \tag{1.45}$$

where

$$\mathbf{L}_\omega(\mathbf{u},\theta) = \mathbf{I}_3 + \frac{\theta}{2}[\mathbf{u}]_\times + \left(1 - \frac{\operatorname{sinc}(\theta)}{\operatorname{sinc}^2(\theta/2)}\right)[\mathbf{u}]_\times^2 \tag{1.46}$$

and $\operatorname{sinc}(\theta) = \sin(\theta)/\theta$

Thus, for PBVS the interaction matrix takes the following form:

$$\mathbf{L_s} = \begin{bmatrix} -\mathbf{I}_3 & [\boldsymbol{e}_t]_\times \\ \mathbf{0}_3 & \mathbf{L}_\omega \end{bmatrix} \tag{1.47}$$

Note that the determinant of $\mathbf{L}_\omega(\mathbf{u},\theta)$ is $\det(\mathbf{L}_\omega) = 1/\operatorname{sinc}^2(\theta/2)$. Thus $\mathbf{L}_\omega(\mathbf{u},\theta)$, and therefore $\mathbf{L_s}$, is singular if and only if $\theta = 2q\pi$ with $q \neq 0$, which is outside of the workspace of the CDPRs under study. The camera velocity is expressed as in (1.33), however for PBVS the inverse of the interaction matrix can be directly used, because here it is a (6×6)–matrix and of full rank [CH08].

Ideally, with the chosen feature vector $\mathbf{s}$, the origin of the object will have a pure straight-line trajectory in the image. Thus, if the origin is defined close to the center of

gravity of the object, it should never leave the image. However, the Cartesian trajectory of the camera in the world frame does not follow a straight line.

If the camera trajectory must be a straight line, it can be achieved by changing the design of the feature vector $\mathbf{s}$. In this case, it is defined as $\mathbf{s} = \begin{bmatrix} {}^{c^*}\mathbf{t}_c & \theta\mathbf{u} \end{bmatrix}^\top$ [CH08], where $\theta\mathbf{u}$ is the axis-angle expression of ${}^{c^*}\mathbf{R}_c$ as mentioned before; and ${}^{c^*}\mathbf{t}_c$ is the translational vector from the desired camera frame to the current one. Since $\mathbf{s}$ already expresses the difference between current and desired camera frames, then $\mathbf{s}^* = \mathbf{0}$ and $\boldsymbol{e} = \mathbf{s} - \mathbf{s}^* = \mathbf{s}$. For this feature vector the interaction matrix becomes:

$$\mathbf{L}_\mathbf{s} = \begin{bmatrix} {}^{c^*}\mathbf{R}_c & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{L}_\omega \end{bmatrix} \tag{1.48}$$

It should be noted that this $\mathbf{L}_\mathbf{s}$ is a block-diagonal matrix, thus there is a decoupling between translational and rotational motions. Furthermore, the interaction matrix depends only on the rotational difference between $\mathcal{F}_c$ and $\mathcal{F}_{c^*}$. Indeed, ${}^{c^*}\mathbf{R}_c$ is a rotation matrix and does not contain any translation information. Similarly, as can be seen in (1.46), $\mathbf{L}_\omega$ only depends on the rotational error.

Even though ideally the camera trajectory in the world frame now is a pure straight line, it is no more true for the origin of the object in the image [CH08]. Some particular configurations could surely be found, in which the object leaves the camera field of view.

1.4.5 2½D Visual Servoing

2½D Visual Servoing (2½D VS) is a hybrid approach that has the advantages of both IBVS and PBVS, while suffering from the disadvantages of neither.

The hybrid nature of this approach lies in the design of the feature vector $\mathbf{s}$. This time it is a combination of 2D and 3D features. More precisely, the current feature vector is defined as $\mathbf{s} = \begin{bmatrix} {}^{c^*}\mathbf{t}_c^\top & x_o & y_o & \theta u_z \end{bmatrix}^\top$ [MCB99] [KKC04]. Here, ${}^{c^*}\mathbf{t}_c$ is the translation vector between the desired camera frame $\mathcal{F}_{c^*}$ and the current camera frame $\mathcal{F}_c$; x_o and y_o are the image coordinates of the object center $\mathbf{o}$; θu_z is the third component of $\theta\mathbf{u}$ vector, where $\mathbf{u}$ is the axis and θ is the angle of the rotation matrix ${}^{c^*}\mathbf{R}_c$. The error vector $\boldsymbol{e}$ is defined by comparing $\mathbf{s}$ to $\mathbf{s}^*$, namely

$$\boldsymbol{e} = \mathbf{s} - \mathbf{s}^* \tag{1.49}$$

It should be noted that all components of the desired feature vector $\mathbf{s}^*$ except x_o^* and y_o^* are equal to 0. Thus, if one would define the desired object center-point coordinate vector $\mathbf{o}^* = [0; 0]^\top$, then $\mathbf{s}^* = \mathbf{0}$ and (1.49) would simplify to $\boldsymbol{e} = \mathbf{s}$.

In perfect conditions, this choice of visual features leads to a straight-line trajectory of the camera (because ${}^{c^*}\mathbf{t}_c$ is part of $\mathbf{s}$), as well as a straight-line trajectory of object center-point $\mathbf{o}$ in the image (as (x_o, y_o) is also part of $\mathbf{s}$). The translational degrees of

freedom are used to realize the 3D straight line of the camera, while the rotational degrees of freedom are devoted to the realization of the 2D straight line of point $\mathbf{o}$.

The interaction matrix $\mathbf{L_s}$ corresponding to the chosen feature vector $\mathbf{s}$ takes the following form for 2½D VS [MCB99] [KKC04] [CH08]:

$$\mathbf{L_s} = \begin{bmatrix} {}^{c^*}\mathbf{R}_c & \mathbf{0}_3 \\ \frac{1}{Z}\mathbf{L}_v & \mathbf{L}_{v\omega} \end{bmatrix} \tag{1.50}$$

with:

$$\mathbf{L}_v = \begin{bmatrix} -1 & 0 & x_o \\ 0 & -1 & y_o \\ 0 & 0 & 0 \end{bmatrix} \tag{1.51}$$

$$\mathbf{L}_{v\omega} = \begin{bmatrix} x_o y_o & -(1+x_o^2) & y_o \\ (1+y_o^2) & -x_o y_o & -x_o \\ l_1 & l_2 & l_3 \end{bmatrix} \tag{1.52}$$

l_1, l_2, l_3 being the components of the third row of matrix $\mathbf{L}_\omega$ in (1.46), and Z being the current Cartesian distance along z axis from the camera to the object.

Finally, camera velocity ${}^c\mathbf{v}_c$ is expressed as in (1.33).

1.4.6 Visual Servoing with a Moving Object

It is important to know whether the object is in motion. Having a moving object means that the error $\boldsymbol{e}$ will change with time, with or without robot motion. This can be expressed as [CH08]:

$$\dot{\boldsymbol{e}} = \mathbf{L_s}\,{}^c\mathbf{v}_c + \frac{\delta \boldsymbol{e}}{\delta t} \tag{1.53}$$

where the term $\frac{\delta \boldsymbol{e}}{\delta t}$ corresponds to the time variation of $\boldsymbol{e}$ due to the object motion.

To compensate the object motion, the computation of the camera velocity ${}^c\mathbf{v}_c$ is changed to:

$${}^c\mathbf{v}_c = -\lambda\, \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\, \boldsymbol{e} - \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\, \widehat{\frac{\delta \boldsymbol{e}}{\delta t}} \tag{1.54}$$

where the first part corresponds to the classic VS, as shown in (1.33), and in the second part the $\widehat{\frac{\delta \boldsymbol{e}}{\delta t}}$ is the estimation of the error variation due to object motion.

The closed-loop equation can be expressed as:

$$\dot{\boldsymbol{e}} = -\lambda\, \mathbf{L_s}\, \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\, \boldsymbol{e} - \mathbf{L_s}\, \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\, \widehat{\frac{\delta \boldsymbol{e}}{\delta t}} + \frac{\delta \boldsymbol{e}}{\delta t} \tag{1.55}$$

According to Lyapunov analysis of the closed-loop equation, the system will remain stable as long as $\mathbf{L_s}\,\widehat{\mathbf{L}}_\mathbf{s}^\dagger > \mathbf{0}$ [Kha02]. However, even if the system is stable, the error $\boldsymbol{e}$ will converge to $\mathbf{O}$ if and only if the estimation $\widehat{\frac{\delta \boldsymbol{e}}{\delta t}}$ is sufficiently accurate so that

$$\mathbf{L_s}\,\widehat{\mathbf{L}}_\mathbf{s}^\dagger\,\widehat{\frac{\delta \boldsymbol{e}}{\delta t}} = \frac{\delta \boldsymbol{e}}{\delta t} \tag{1.56}$$

Otherwise tracking errors will be observed. This can be explained by a simple example from [CH08], where a scalar differential equation $\dot{e} = -\lambda e + b$, which is a simplification of (1.55), is analyzed. The solution is $e(t) = e(0)\exp(-\lambda t) + b/\lambda$, which converges towards b/λ. Increasing λ reduces the tracking error. However, if it is too high, it can yield the system unstable. Therefore, it is necessary to keep b as small as possible.

If the object motion is unknown, which is true most of the time, then one approach to estimate the object motion is given by:

$$\widehat{\frac{\delta \boldsymbol{e}}{\delta t}} = \frac{\mathbf{s}(t) - \mathbf{s}(t-\delta t)}{\delta t} - \mathbf{L_s}(t-\delta t)\,{}^{c}\mathbf{v}_c(t-\delta t) \tag{1.57}$$

where the feature vector at time t is compared to its previous value at time $(t-\delta t)$. A Kalman filter [CG93] can then be used to improve the estimated values obtained. Details on other estimation approaches are given in [CH08].

Choice of Visual Features for the Tracking Task

For some tasks it is not necessary to control all the degrees of freedom of the moving-platform. For example, in [RCM14] the task was to retrieve objects from the environment using a magnet and the orientation of the object during the picking and the placing operations was not important. Similarly, when picking a spherical object, the orientation of the end-effector about its z axis can remain unconstrained.

As part of our experimental validation, we define a tracking task, where the object is executing a full circular path. As no manipulation of the object is needed, tracking of the object orientation about the global z axis is not necessary and thus the last DoF of the moving-platform can remain free. To do so, the feature vector $\mathbf{s}$ needs to be adapted to ignore the Z component of object orientation.

As described in Section 1.4.4, typically for one choice of PBVS the feature vector has the following composition: $\mathbf{s} = ({}^{c}\mathbf{t}_o, \theta\mathbf{u})$, where ${}^{c}\mathbf{t}_o$ corresponds to the object position expressed in camera frame, and $\theta\mathbf{u}$ is the axis-angle representation of the rotation matrix ${}^{c^*}\mathbf{R}_c$ between the desired and the current camera frames. To avoid tracking object orientation about z axis, θu_z is removed from the feature vector $\mathbf{s}$. Thus, only five features remain and only five degrees of freedom of the moving-platform are controlled. It means that the moving-platform orientation about z axis is free.

Similarly, adapting 2½D VS feature vector to track an object moving in a circular trajectory simply means removing the last feature from its feature vector $\mathbf{s}$ so that it takes

the form $\mathbf{s}_5 = ({}^{c^*}\mathbf{t}_c, x_o, y_o)$. Here, ${}^{c^*}\mathbf{t}_c$ is the translation vector between the desired and the current camera frames, and (x_o, y_o) are the coordinates of the object center-point in the image.

In case of IBVS a completely different set of features can be taken. Instead of using four points as described in Section 1.4.3, two centered concentric circles parallel to the image plane can be observed [CRE93]. The pattern is shown in Fig. 1.23. Each circle is represented by an ellipse in the image plane and the corresponding feature vector for an ellipse is defined as $\mathbf{s} = (x_c, y_c, n_{20}, n_{11}, n_{02})$, where (x_c, y_c) are the normalized coordinates of the ellipse center; n_{20}, n_{11}, n_{02} are normalized centered moments [Cha04]. In this case, the interaction matrix related to the two centered concentric circles parallel to the image plane is:

$$\mathbf{L}_{cc} = \begin{bmatrix} -\frac{1}{Z_c} & 0 & 0 & 0 & -1-R_1^2 & 0 \\ 0 & -\frac{1}{Z_c} & 0 & 1+R_1^2 & 0 & 0 \\ 0 & 0 & \frac{2R_1^2}{Z_c} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{2R_1^2}{Z_c} & 0 & 0 & 0 \\ -\frac{1}{Z_c} & 0 & 0 & 0 & -1-R_2^2 & 0 \\ 0 & -\frac{1}{Z_c} & 0 & 1+R_2^2 & 0 & 0 \\ 0 & 0 & \frac{2R_2^2}{Z_c} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{2R_2^2}{Z_c} & 0 & 0 & 0 \end{bmatrix} \tag{1.58}$$

where R_1 is the radius of the first circle; R_2 is the radius of the second circle; and Z_c is the distance along z axis of the circle plane from $\mathcal{F}_c$. Furthermore, as two circles are used, the rank of matrix $\mathbf{L}_{cc}$ is 5.

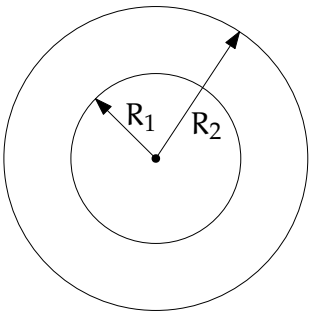


Figure 1.23: Two concentric circle pattern

Note that the last column of the interaction matrix gives the expected non-controlled 3D motion. Indeed, the rotation of the pattern or of the camera about the camera optical axis is invisible. Thus a virtual revolute link between the camera and the chosen pattern is created [CRE93].

1.4.7 Visual Servoing with Trajectory Planning and Tracking

It is well known that having perturbations in the system, which do not cause loss of stability, has an undesirable effect on the trajectory. Trajectory planning and following can be used to increase the robustness of the chosen control w.r.t. modeling errors [MC02] and to preserve the ideal shape of the trajectory [BMG97].

Indeed, the larger the error $\boldsymbol{e} = \mathbf{s} - \mathbf{s}^*$, the bigger the effect of modeling errors on system behavior. When tracking a chosen trajectory, at each iteration the error becomes $\boldsymbol{e}(t) = \mathbf{s}(t) - \mathbf{s}^*(t)$. Consequently, when $t = 0\,s$ we have $\mathbf{s}^*(0) = \mathbf{s}(0)$. Since $\mathbf{s}^*(t)$ is now time varying, the control scheme needs to be slightly changed. More precisely, instead of (1.32) we now have [CH08] [BMG97]:

$$\dot{\boldsymbol{e}} = \dot{\mathbf{s}} - \dot{\mathbf{s}}^* = \mathbf{L}_{\mathbf{s}}\,{}^{c}\mathbf{v}_c - \dot{\mathbf{s}}^* \tag{1.59}$$

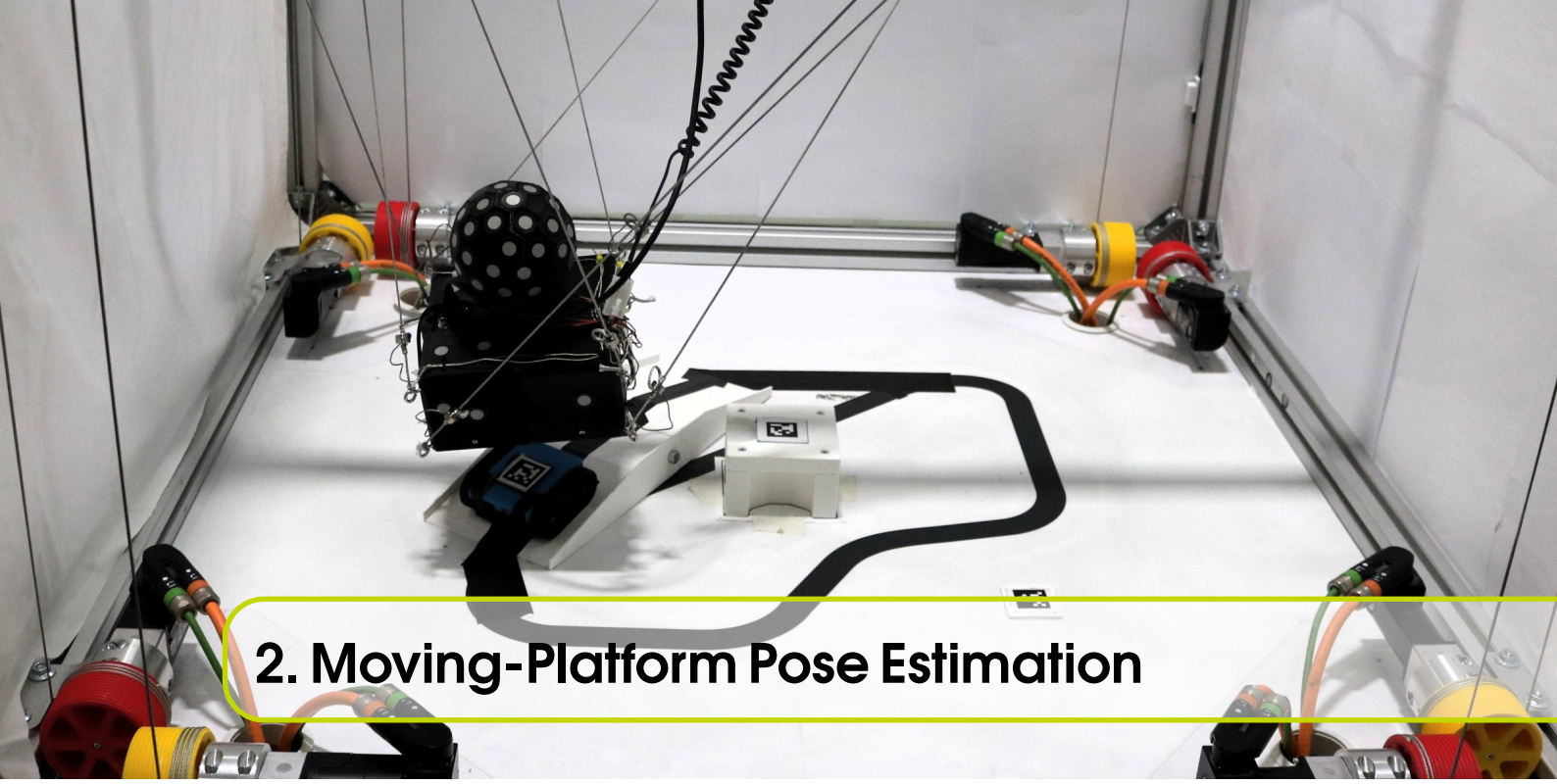
And similarly, to compensate the time variation of the desired feature vector, the camera velocity is now expressed as:

$${}^{c}\mathbf{v}_c = -\lambda\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\boldsymbol{e} - \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\widehat{\dot{\mathbf{s}}}^* \tag{1.60}$$

It should be noted that unlike tracking of mobile objects, described in the previous section, here the velocity $\dot{\mathbf{s}}^*$ is known. Indeed, it is set during the trajectory planning phase. Thus, $\widehat{\dot{\mathbf{s}}}^* = \dot{\mathbf{s}}^*$ and (1.60) becomes:

$${}^{c}\mathbf{v}_c = -\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\left(\lambda\boldsymbol{e} + \dot{\mathbf{s}}^*\right) \tag{1.61}$$

The success of any trajectory tracking is based on the time available to complete the task. The higher the trajectory time t_{full}, the more accurate the trajectory tracking. Indeed, the larger t_{full}, the lower the robot velocity, and the smaller the path step between two iterations. This leads to a smaller difference between $\mathbf{s}^*(t)$ and $\mathbf{s}^*(t + \delta t)$, which in turn means a smaller difference between $\mathbf{s}(t)$ and $\mathbf{s}^*(t + \delta t)$, thus a better path following. On the other hand, we do not want the task execution to be too slow, thus a compromise has to be found.



2. Moving-Platform Pose Estimation

Contents

2.1 Introduction

When the only camera in the system is an onboard camera, it is not possible to observe the moving-platform pose. To be able to compute the required cable velocities $\dot{l}$ with (1.39), the knowledge of current moving-platform pose is necessary. More precisely, (1.39) contains the kinematic Jacobian matrix $\mathbf{A}$, which depends on the moving-platform pose as can be seen from (1.19) and (1.4).

Thus, it is necessary to estimate the moving-platform pose. Furthermore, the quality of this estimation can influence the behavior of the robot. Therefore, it is necessary to ensure that a good estimation method is used.

In this chapter several moving-platform estimation methods are developed. Extensive experimental validations on two CDPRs and on different tasks are performed to find the most versatile and robust estimation method.

2.2 Moving-Platform Pose Estimation Methods

Three moving-platform pose estimation methods are presented in this section, namely: (i) control-based; (ii) image-based; (iii) model-based. In all three cases it is assumed that the initial moving-platform pose ${}^{b}\mathbf{p}_{p_0}$ at $t = 0\,s$ is known.

The proposed methods do not depend on the chosen visual servoing approach. Indeed, in any case, the control scheme remains as shown in Fig. 1.22. It does not matter whether the controller is extended with trajectory planning and tracking, the principle of the moving-platform pose estimation methods remains the same. However, special care is necessary for mobile object tracking, because only the control-based and model-based methods can be used in that case.

2.2.1 Control-Based Estimation

Assuming that the velocity produced by the CDPR corresponds to the velocity prescribed by the controller, the control output can be used for the moving-platform pose estimation. More precisely at every iteration the homogeneous transformation matrix ${}^{b}\mathbf{T}_{p}$ can be updated as follows:

$$ {}^{b}\mathbf{T}_{p}(t) = {}^{b}\mathbf{T}_{p}(t - \Delta t)\, e^{({}^{p}\mathbf{v}_{p}, \Delta t)} \tag{2.1} $$

where $e^{({}^{p}\mathbf{v}_{p}, \Delta t)}$ is the exponential map [Ead14] given a velocity ${}^{p}\mathbf{v}_{p}$ and a time interval Δt. Note that ${}^{p}\mathbf{v}_{p}$ is computed by the controller.

2.2.2 Image-Based Estimation

The features from the previous image are recorded and then compared to the current features. The difference in features allows us to retrieve the camera displacement between the frames and thus the new moving-platform pose can be computed.

Assuming that 3D features are available, like in PBVS and 2½D VS, it is possible to express the object pose in the camera frame $\mathcal{F}_c$ as a homogeneous transformation matrix ${}^{c}\mathbf{T}_o$. It can then be expressed in the base frame $\mathcal{F}_b$ as:

$$ {}^{b}\mathbf{T}_o = {}^{b}\mathbf{T}_p \, {}^{p}\mathbf{T}_c \, {}^{c}\mathbf{T}_o \tag{2.2} $$

The transformation matrices present in (2.2) and the corresponding frames are shown in Fig. 2.1.

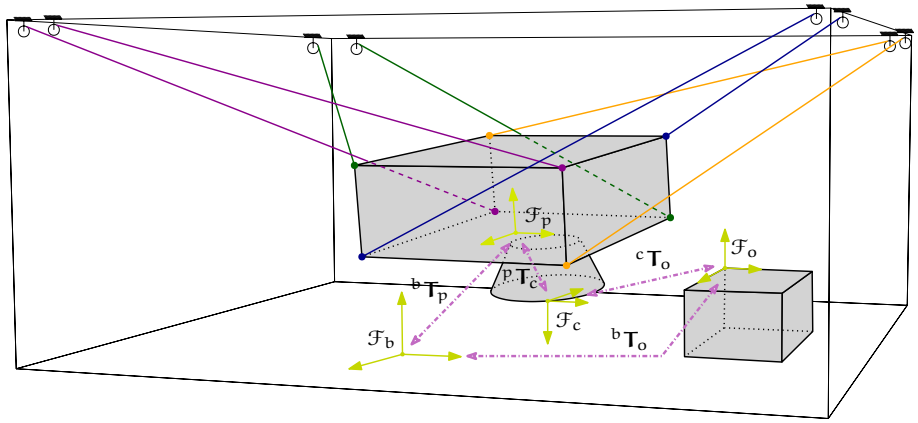


Figure 2.1: Schematic of a spatial CDPR with the relevant homogeneous transformation matrices

Taking into account that the camera is fixed on the moving-platform and thus ${}^{p}\mathbf{T}_c$ does not change with time, at two different iterations (2.2) can be expressed as:

$$ \begin{cases} {}^{b}\mathbf{T}_o(t-\Delta t) = {}^{b}\mathbf{T}_p(t-\Delta t) \, {}^{p}\mathbf{T}_c \, {}^{c}\mathbf{T}_o(t-\Delta t) \\ {}^{b}\mathbf{T}_o(t) = {}^{b}\mathbf{T}_p(t) \, {}^{p}\mathbf{T}_c \, {}^{c}\mathbf{T}_o(t) \end{cases} \tag{2.3} $$

where Δt is the time period between two iterations.

For a motionless object, ${}^{b}\mathbf{T}_o$ does not change with time, thus ${}^{b}\mathbf{T}_o(t-\Delta t) = {}^{b}\mathbf{T}_o(t)$ and we can express ${}^{b}\mathbf{T}_p(t)$ from (2.3):

$$ {}^{b}\mathbf{T}_p(t) = {}^{b}\mathbf{T}_p(t-\Delta t) \, {}^{p}\mathbf{T}_c \, {}^{c}\mathbf{T}_o(t-\Delta t) \, {}^{c}\mathbf{T}_o^{-1}(t) \, {}^{p}\mathbf{T}_c^{-1} \tag{2.4} $$

Thus, in the following experiments (2.4) is used when estimating the new moving-platform pose from two consecutive images.

Furthermore, as the object is assumed to be motionless, ${}^{b}\mathbf{T}_o(0)$ at $t = 0\,s$ can be computed from (2.2) and then simply used as a known and constant value in:

$$ {}^{b}\mathbf{T}_p(t) = {}^{b}\mathbf{T}_o(0) \, {}^{c}\mathbf{T}_o^{-1}(t) \, {}^{p}\mathbf{T}_c^{-1} \tag{2.5} $$

In the following experiments, (2.5) is used to estimate the new moving-platform pose from the first and the current image. Note that here the initial moving-platform pose is still required to be able to compute the homogeneous transformation ${}^{b}\mathbf{T}_{o}(0)$ with (2.2).

Unlike control integration, this approach does not require the CDPR to achieve the control velocity. Indeed, the calculation is done without taking into account the velocity and time spent to achieve the current pose. Instead two separate object pose measurements are used.

Finally, this method can be adapted to indirectly measure the moving-platform pose given that the environment contains a known object at a known location. Indeed, if ${}^{b}\mathbf{T}_{o_k}$ is known, then ${}^{b}\mathbf{T}_{p}$ can be directly expressed as ${}^{b}\mathbf{T}_{p} = {}^{b}\mathbf{T}_{o_k}\,{}^{c}\mathbf{T}_{o_k}^{-1}\,{}^{p}\mathbf{T}_{c}^{-1}$. This method is not used in the following experiments, because it requires adding supplementary objects in the environment that ideally would need to be seen throughout the experiment.

Tracking a Moving Object

When tracking a moving object, the difference between two images is due to both the camera motion and the object motion. Usually, the object motion is unknown. Because of this, it is impossible to use this estimation method.

It should be noted that any target object displacement will lead to a wrong moving-platform pose estimation. This also applies to an accidental shifting of the target. Indeed, as the object motion is not known and, more importantly, not expected, any and all target object pose changes with respect to the camera frame are assumed to be due to the moving-platform displacement.

2.2.3 Model-Based Estimation

The Direct Geometric Model (DGM) expresses the moving-platform pose as a function of cable lengths. For a spatial CDPR with 8 cables this is a very complex problem, which often cannot be solved rapidly enough to be used in control. Special approaches such as interval analysis have been used in [Ber15] [BMC16] [MADS15] to obtain all of the solutions for the Direct Geometrico-Static Model (DGSM), however this usually comes at a relatively high computational cost. Furthermore, there is the need to then determine, which solution is the correct one.

Using cable tension measurements, it is possible to estimate the moving-platform pose given a previous pose and cable lengths. Several approaches are presented in the following sections, depending on the selected CDPR model complexity.

Simplest CDPR model

In the simplest CDPR model the cables are assumed to be massless, non-elastic and always straight. In such a case, at any given time at most six cables can be in tension [Mer17]. Thus, the DGM can be simplified to only take into account the six cables that are most in tension. Such a model is similar to the Gough-Stewart platform, whose

DGM can have up to 40 solutions. The computation of all of the solutions can be a lengthy process, which is not necessary in this case. Indeed, we are only interested in the solution that is in the vicinity of the last known pose. In this case, it is possible to use the least squares optimization method to find the moving-platform pose [Pot18]. The estimation procedure is shown in Algorithm 2.

Algorithm 2: Moving-platform pose estimation from cable lengths

Initialization
  Define the CDPR model with the Cartesian coordinates of A_i and B_i expressed in $\mathcal{F}_b$ and $\mathcal{F}_p$, respectively.
  Define the initial moving-platform pose and record it in ${}^b\mathbf{p}_{p_o}$
End of Initialization
Pose Estimation
  while True **do**
    Get measurements: cable tensions $\boldsymbol{\tau}$ and lengths $\mathbf{l}$
    Find the six largest cable tensions
    Compose the residual equations for these six cables: $f_i = \|{}^b\mathbf{b}_i - {}^b\mathbf{a}_i\|_2 - l_i$
    Compose the equation to normalize the quaternions: $f_7 = q_1^2 + q_2^2 + q_3^2 + q_4^2 - 1$
    Find ${}^b\mathbf{p}_p$ with the least squares method, giving ${}^b\mathbf{p}_{p_o}$ as the initial guess and f_1 to f_7 as the system to be minimized
    Update ${}^b\mathbf{p}_{p_o} = {}^b\mathbf{p}_p$ for next iteration
  end while
End of Pose Estimation

Note that here the expression of the moving-platform pose is important. For example, by expressing it as the homogeneous transformation matrix, there are 12 unknowns for 6 equations. Instead, the pose can be expressed as ${}^b\mathbf{p}_p = \begin{bmatrix} x & y & z & q_1 & q_2 & q_3 & q_4 \end{bmatrix}$, where $\begin{bmatrix} x & y & z \end{bmatrix}$ is the translational part and $\begin{bmatrix} q_1 & q_2 & q_3 & q_4 \end{bmatrix}$ is the rotational part expressed as quaternions. Note that, when using quaternions, their normalizing equation $q_1^2 + q_2^2 + q_3^2 + q_4^2 = 1$ must also be taken into account. Thus, there are 7 unknowns and 7 equations, which makes this system solvable.

The least squares algorithm to compute the moving-platform pose is fast (for example the implementation in Python using *scipy.least_squares* takes about 0.009 s). Furthermore, using only the lengths of six cables that are most in tension makes the algorithm robust to cable slackness, because the slack cables are ignored. However, it is a model-based estimation method, meaning that any error in the model or in the initial moving-platform pose will lead to moving-platform pose estimation errors.

Elastic cables

Cable elasticity affects the final moving-platform pose as well as the tensions that are obtained in the cables. Indeed, only with elastic cables it is possible to have all eight of them in tension [Mer17].

When considering elasticity, the ith cable tension at time t can be expressed as:

$$\tau_i(t) = \mathcal{K}(t)\,\delta l_i + \tau_i(0) \tag{2.6}$$

where:

- δl_i is the elongation of the ith cable;
- $\tau_i(0)$ is the ith cable tension with the moving-platform positioned on the ground;
- $\mathcal{K}(t)$ is the cable stiffness, which is expressed as $\mathcal{K}(t) = ES/l_i(t)$;
- E is the Young modulus of the ith cable;
- S is the cross-section of the ith cable;
- $l_i(t)$ is the length of the ith cable at time t.

If tension measurements are available, then it will be possible to compute the cable elongation from (2.6):

$$\delta l_i = \frac{\big(\tau_i(t) - \tau_i(0)\big) l_i(t)}{ES} \tag{2.7}$$

Thus the ith cable length l_i obtained from the motor encoders can be corrected to $\widehat{l}_i = l_i + \delta l_i$. The estimation of the moving-platform pose is done as described in Algorithm 2, but in the equation on line 9 the cable length l_i is substituted by $\widehat{l}_i$.

Pulley kinematics

Pulley kinematics have been introduced in Section 1.3.1. The only change to Algorithm 2 is on line 9. Indeed, we no longer need the straight line length from A_i to B_i, instead it is the cable length from point A_i' on the pulley to B_i. Thus on line 9 the equation (1.5) is used instead.

Elastic cables and pulley kinematics

Finally, we can also combine pulley kinematics and cable elasticity in the moving-platform pose estimation. The full cable length from A_i to B_i that will be used in Algorithm 2 on line 9 becomes:

$$\hat{l}_{fi} = \widehat{L}_i + l_{pi} = L_i + \delta L_i + l_{pi} \tag{2.8}$$

where L_i, l_{pi} and δL_i are computed by (1.16), (1.6), and (2.7) respectively.

2.3 Case Study I: Evaluation of Moving-Platform Pose Estimation Methods

To evaluate the different estimation methods, several experiments were performed on both CDPR prototypes (see Appendix A.1). The estimation methods are defined as follows:

- Est. 1 is the estimation using control integration;
- Est. 2 is the estimation using two consecutive images;
- Est. 3 is the estimation using the first and current image;

- Est. 4 is the estimation from cable lengths;
- Est. 5 is the estimation from cable lengths taking cable elasticity into account;
- Est. 6 is the estimation from cable lengths taking pulley geometry into account;
- Est. 7 is the estimation from cable lengths taking cable elasticity and pulley geometry into account;
- Meas. 1 (for ACROBOT only) is the method where measurements from HTC Vive are used inside the control instead of estimating the moving-platform pose.

Note that measurement by the HTC Vive (presented in Appendix A.1) was only possible on ACROBOT due to the size of CAROCA. Creaform C-Track measurement system, presented in Appendix A.2, was used to get the ground truth measurement of the moving-platform pose for all experiments for off-line comparison. It was also used to calibrate the HTC Vive tracker with the moving-platform at the center of the workspace.

First, an evaluation with an open-loop controller is done in Section 2.3.1. Here, the position controller described in Appendix A.4 is used.

Then, the PBVS control presented in Section 1.4.4 is used in Sections 2.3.2 to 2.3.5. Indeed, we are interested in finding the estimation method best suited for visual servoing, thus dedicated experiments were necessary. More precisely, for the PBVS controller the feature vector $\mathbf{s}$ is defined as $\mathbf{s} = (\mathbf{s}_t, \mathbf{s}_\omega)$, where $\mathbf{s}_t = {}^c\mathbf{t}_o$ is the translational distance between the camera frame and the object frame; and $\mathbf{s}_\omega = \theta\mathbf{u}$ is the axis-angle representation of the rotational matrix ${}^{c^*}\mathbf{R}_c$ between the desired and the current camera frames. As a simplification of the computer vision part, fiducial markers named AprilTags [Ols11] are used as objects. They are recognized and localized by algorithms available in the ViSP library [MSC05].

Adaptive gain λ, defined in (1.31), is used. For ACROBOT the coefficients have been tuned at the following values: $\lambda_0 = 2.0$, $\lambda_\infty = 0.3$ and $\dot{\lambda} = 30$. On the other hand, for CAROCA the tuned values are: $\lambda_0 = 1.5$, $\lambda_\infty = 0.1$ and $\dot{\lambda} = 30$. For both CDPRs the IDS camera is mounted on the moving-platform to observe the ground and the AprilTags located on it.

2.3.1 Open-loop velocity control on ACROBOT

A simple open-loop controller described in Appendix A.4 is implemented to generate a straight-line trajectory between the following points:

- ${}^b\mathbf{p}_{p0} = [0.112\,\text{m}, -0.032\,\text{m}, 0.301\,\text{m}, -12°, -10°, 0°]$
- ${}^b\mathbf{p}_{p1} = [0.284\,\text{m}, -0.197\,\text{m}, 0.08\,\text{m}, 0°, 0°, 0°]$
- ${}^b\mathbf{p}_{p2} = [0.284\,\text{m}, -0.197\,\text{m}, 0.25\,\text{m}, 0°, 0°, 0°]$

Initially the moving-platform is at ${}^b\mathbf{p}_{p0}$ and a trajectory is generated to reach ${}^b\mathbf{p}_{p1}$; then height along z axis is increased to reach ${}^b\mathbf{p}_{p2}$; and finally the moving-platform returns to ${}^b\mathbf{p}_{p0}$.

For the image-based estimation method we use an AprilTag as an object. It is placed so that it is visible throughout the experiment.

The experimental results are shown in Figs. 2.2 and 2.3[2]. The task was executed once using the controller described above. The planned moving-platform trajectory is shown in orange in Fig. 2.2a. The moving-platform trajectory was measured by C-Track and it is shown in cyan in Fig. 2.2a. Note that while the initial pose ${}^{b}\mathbf{p}_{p0}$ is indeed where the experiment starts, however the poses ${}^{b}\mathbf{p}_{p1}$ and ${}^{b}\mathbf{p}_{p2}$ are not reached accurately. At the end of the first part of the trajectory the moving-platform passes ${}^{b}\mathbf{p}_{p1}$ and comes to a stop right afterwards. Then only the z component is changed, thus ${}^{b}\mathbf{p}_{p2}$ is also not achieved accurately. Finally, the moving-platform accurately returns to the initial pose ${}^{b}\mathbf{p}_{p0}$. Indeed, as can be seen in Figs. 2.3a and 2.3d the initial and final images received from the camera are identical. Thus, it can be concluded that the CDPR executes the control velocity $\dot{\mathbf{l}}$ precisely. The positioning error in ${}^{b}\mathbf{p}_{p1}$ and ${}^{b}\mathbf{p}_{p2}$ can also be seen in Fig. 2.3. Ideally with the moving-platform in ${}^{b}\mathbf{p}_{p1}$ the AprilTag should be centered in the image and with no orientation about z axis. However, as can be seen in Fig. 2.3b the AprilTag is not centered and it is rotated. Similarly, in Fig. 2.3c the AprilTag is also rotated. These positioning errors come from the fact that the analytical form of the Jacobian matrix $\mathbf{A}$ used in (A.21) is based on a too simplified CDPR model not corresponding to the real robot, due to calibration errors and approximations in the model, e.g., errors in the cable anchor and exit point coordinates, uncertainties in cable elongation, etc.

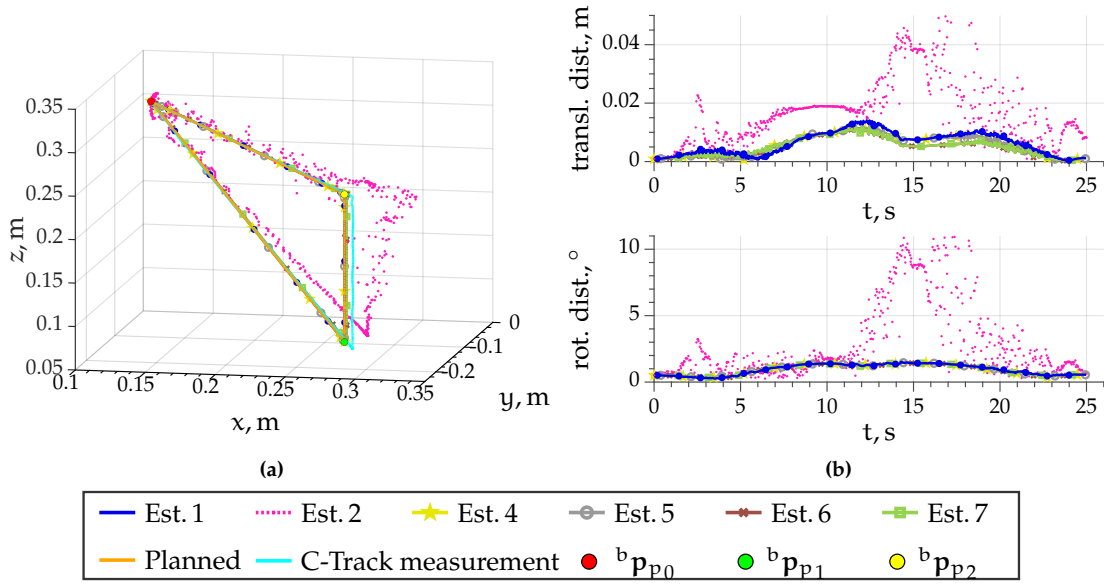


Figure 2.2: Experimental results: **(a)** moving-platform trajectory as estimated by the different estimation methods and as measured by C-Track; **(b)** deviation of the moving-platform pose estimation from the measurement by C-Track

After executing the task, the recorded values were supplied to all moving-platform pose estimation methods. The resulting trajectories are shown in Fig. 2.2 as Est. 1 to Est. 6.

[2]Please also refer to the accompanying video at `https://youtu.be/LAqBTIMj4NI`

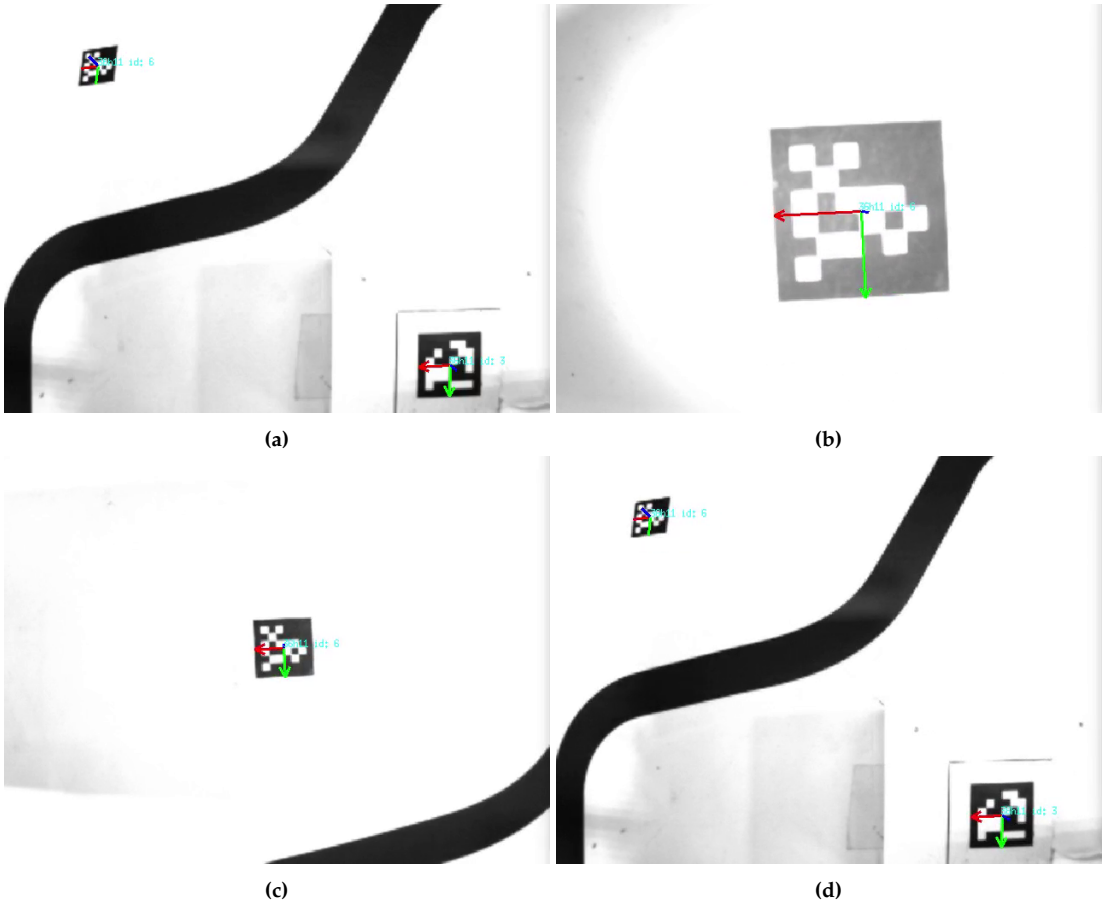


Figure 2.3: Images from the onboard camera: **(a)** at $t = 0\,s$ at ${}^{b}\mathbf{p}_{p0}$; **(b)** at $t = 10\,s$ when ${}^{b}\mathbf{p}_{p1}$ is assumed to be reached; **(c)** at $t = 15\,s$ when ${}^{b}\mathbf{p}_{p2}$ is assumed to be reached; **(d)** at $t = 25\,s$ when ${}^{b}\mathbf{p}_{p0}$ is assumed to be reached.

Surprisingly, the estimated trajectory with control-based and model-based estimators corresponds to the planned one (but not to the achieved one). For instance, at the end of the first part of the trajectory, the estimated moving-platform pose corresponds to ${}^{b}\mathbf{p}_{p1}$, even though the moving-platform is actually shifted by approximately 1 cm. Similarly, at the end of the second part of the trajectory the estimated moving-platform pose corresponds to ${}^{b}\mathbf{p}_{p2}$. This means that all sampling effects are fully negligible in implementing (A.21), leading the robot to achieve the computed velocity ${}^{b}\mathbf{v}_{p}(t)$ and reaching the desired cable length values by successive integration of $\dot{l}$. Finally, as the model-based methods rely on a model, it is not surprising its results deviate from the ground-truth. Similarly, as the control-based method is not based on any measurement, it is not surprising its results also deviate from the ground-truth.

Fig. 2.2b allows us to evaluate the accuracy of estimation with each method. Here, the deviation of the estimation from the actual moving-platform pose measurement is shown as the translational and the rotational distance. Note that the rotational distance is the angle θ from the axis-angle expression of the rotation matrix between the measured and the estimated moving-platform poses. The vertical dotted lines at $t = 10\,s$ and $t = 15\,s$

correspond to the end of the first and second parts of the trajectory, when ${}^{b}\mathbf{p}_{p_1}$ and ${}^{b}\mathbf{p}_{p_2}$ are supposed to be reached.

Let us begin with Est. 2, because it is the only one producing a significantly different result. Est. 2 corresponds to the image-based moving-platform pose estimation method. It is clear that the AprilTag used in this experiment was too small to produce a good measurement. Indeed, note the good accuracy of the measurement between $t = 8\,s$ and $t = 12\,s$, when the moving-platform is the closest to the AprilTag. On the other hand, as the AprilTag becomes significantly smaller in the image, the estimation of the moving-platform pose becomes highly noisy. Thus, with the image-based method the accuracy of the moving-platform pose estimation depends on the distance between the camera and the object and thus its size in the image.

The control-based estimation Est. 1 is shown in blue. The translational deviation reaches 1.4 cm at $t = 12.4\,s$ and the task is finished with a deviation of only 1.3 mm. The estimation of the moving-platform orientation does not surpass 1.6° error and the task is finished with a 0.55° error. The result with the model-based methods is almost the same. More precisely, the largest translational deviation is 1.21 cm at $t = 11.6\,s$ and at the end of the task reduces to just 1 mm, because the moving-platform returns to its initial pose ${}^{b}\mathbf{p}_{p_0}$. The largest rotational distance is 1.6° at $t = 14.3\,s$ and it reduces to 0.55° by the end of the task. Note that the results with all four model-based approaches are almost identical. Indeed, for a small CDPR with Dyneema cables that are almost inelastic and very small pulleys of 9 mm in diameter, the output of the four model-based estimation methods is almost the same.

Finally, it is important to note that the current experimental setup on ACROBOT is almost perfect, that is, its low-level controller is able to achieve the computed velocities. Thus, it would be of interest to extend this study by implementing the same controllers and estimation methods on larger CDPRs, such as CAROCA, with a high payload that are not able to ensure this nice property. Unfortunately, CAROCA was not available during this study.

2.3.2 Approaching a Static Object on ACROBOT by PBVS

The initial and desired values are the following:

- initial state:
  - ◇ ${}^{b}\mathbf{p}_{p_0} = [0.112\,m;\ -0.032\,m;\ 0.301\,m;\ -12°;\ -10°;\ 0°]$
  - ◇ ${}^{c}\mathbf{p}_{o0} = [-0.123\,m;\ -0.101\,m;\ 0.338\,m;\ 168°;\ -8°;\ -179°]$
- desired state:
  - ◇ ${}^{b}\mathbf{p}_{p^*} = [0.284\,m;\ -0.197\,m;\ 0.08\,m;\ 0°;\ 0°;\ 0°]$
  - ◇ ${}^{c}\mathbf{p}^{*}_{o} = [0\,m;\ 0\,m;\ 0.09\,m;\ 180°;\ 0°;\ 180°]$

where if the moving-platform is in pose ${}^{b}\mathbf{p}_{p0}$ with respect to the base frame $\mathcal{F}_b$, the object will be in pose ${}^{c}\mathbf{p}_{o0}$ with respect to the camera frame $\mathcal{F}_c$. Similarly, by positioning the moving-platform in ${}^{b}\mathbf{p}_{p^*}$, the object arrives at ${}^{c}\mathbf{p}^*_o$.

Note that ${}^{b}\mathbf{p}_{p0}$ was measured by Creaform C-Track and then used as the first known pose, while all the following moving-platform poses were estimated by one of the estimation methods. Indeed, a separate experiment was done with each of the moving-platform pose estimation methods, unlike Section 2.3.1.

The same trajectory was repeated five times with each of the estimation methods. The first run with each estimation method is shown in Fig. 2.4, while the average and maximum deviations over all runs are shown in Fig. 2.5. As can be seen in Fig. 2.4a the AprilTag center-point trajectory is almost ideal in all cases. Indeed, the pixel deviation from the ideal straight-line trajectory does not surpass 15 pixels (see Fig. 2.4b). Moreover,

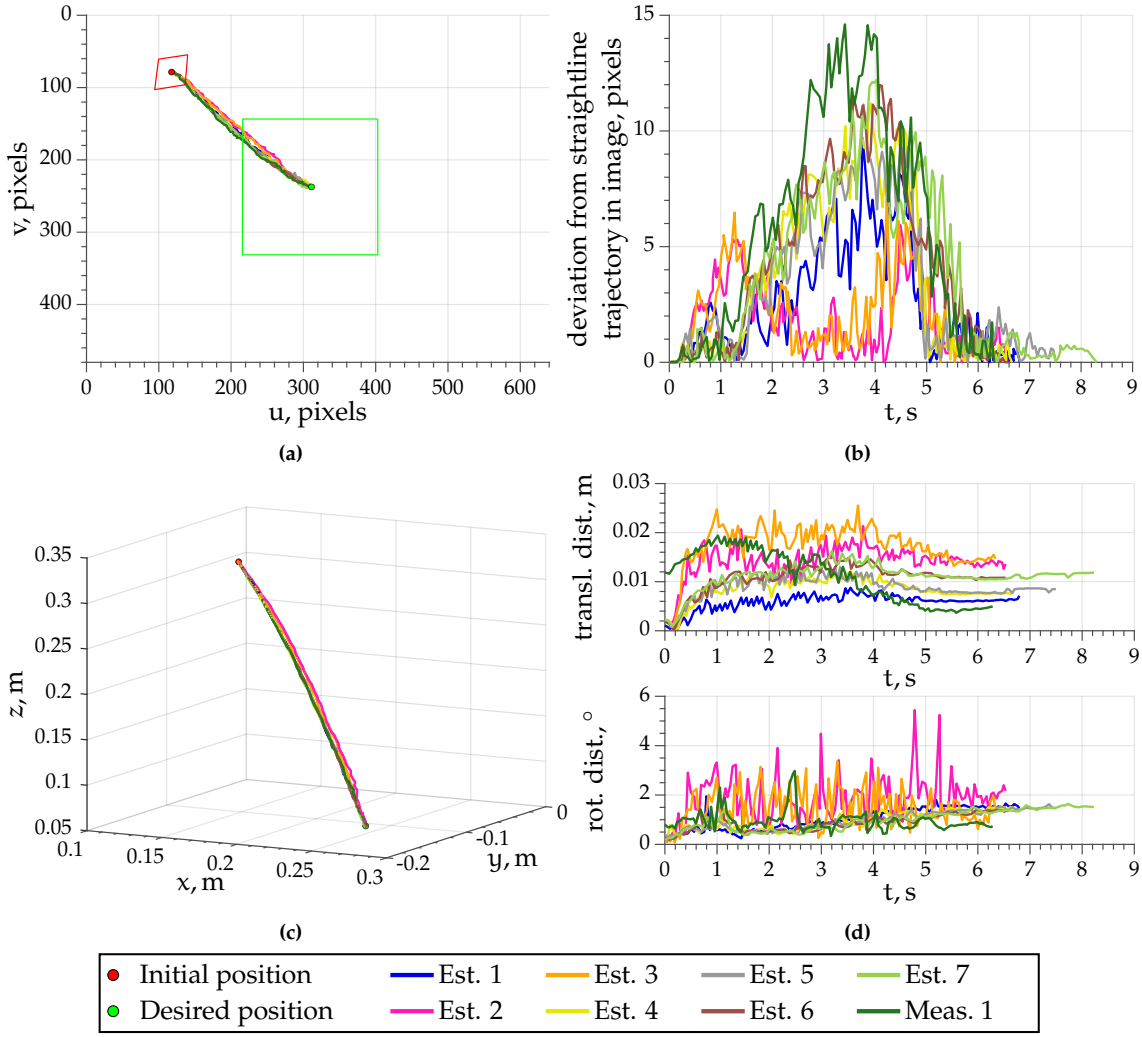


Figure 2.4: Estimation method comparison on ACROBOT with a static object: **(a)** AprilTag center-point trajectory in the image; **(b)** deviation from the ideal straight-line trajectory in the image; **(c)** moving-platform trajectory in the base frame; **(d)** deviation of the moving-platform pose estimation with respect to C-Track measurement

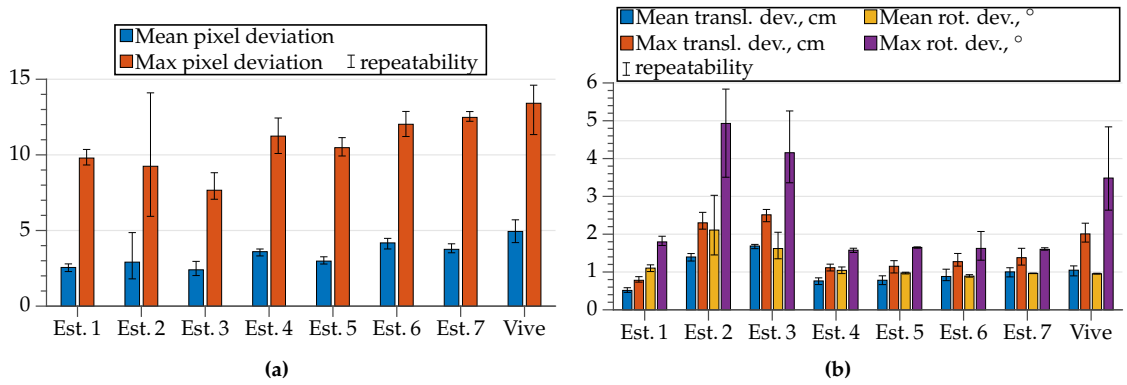


Figure 2.5: Quality of the estimation methods on ACROBOT: **(a)** deviation of the AprilTag center-point in the image; **(b)** moving-platform pose estimation deviation with respect to C-Track measurement

for the control-based and image-based estimation approaches, namely Est. 1, 2 and 3, the deviation does not surpass 10 pixels. Similarly, the moving-platform trajectories, which were measured by Creaform C-Track and are shown in Fig. 2.4c, are also almost identical no matter the estimation method.

Moving-platform pose estimation deviation from the C-Track measurement is shown in Fig. 2.4d. Here, the translational distance and the angle θ of the axis-angle distance are shown. The distance is computed by taking the C-Track measurement as ground truth and comparing it to each estimation output at every time instant. The estimation by control integration, shown in blue, appears to have the lowest translational deviation. It is closely followed by model-based estimation approaches. Note that there is almost no difference between Est. 4 and Est. 5, as well as between Est. 6 and Est. 7. Thus, it appears that taking into account cable elasticity does not give any advantage, which is not surprising given that the chosen Dyneema cables are quasi inelastic. However, interestingly, the pair of estimation methods with pulley kinematics, namely Est. 6 and Est. 7, appear to have a slightly larger final estimation deviation. Possibly, the modeling of ACROBOT pulleys needs to be improved to take into account a small rotation that is possible around its attachment point shown in Fig. 1.13. Finally, for the two image-based approaches Est. 2 and Est. 3 the estimation deviation is the largest one and also it appears to be the noisiest one. It should be noted that it is not surprising to have a worse estimation at the beginning of the trajectory, because initially the AprilTag is very small in the image, as can be seen in Fig. 2.4a. Indeed, as the robot approaches the desired state, the AprilTag becomes larger and the estimation of the moving-platform pose becomes less noisy. However, the estimation deviation at the end of the trajectory remains the largest, compared to other estimation methods.

Regarding the moving-platform orientation estimation, the control-based and model-based methods produce basically the same deviation of less than 2°. On the other hand, the image-based methods are noisy with spikes up to almost 6°.

Regarding the experiment repeatability, Est. 1 appears to give the best results. Indeed, while Est. 2 and Est. 3 produce smaller pixel deviation spikes in Fig. 2.5a, the repeatability is worse, especially for Est. 2. All model-based methods and HTC Vive measurement produce worse results. Est. 2 produces the worst repeatability for moving-platform pose estimation, as shown in Fig. 2.5b, especially in rotation. It is closely followed by Est. 3 and Meas. 1. Model-based methods all produce similar results, however it appears that the simplest model used in Est. 4 is the best suited for ACROBOT. Comparing Est. 1 to Est. 4, the former produces a better result for translation estimation, while the latter results in a slightly better orientation estimation.

Note that the use of HTC Vive in Meas. 1 does not appear to improve the behavior of the CDPR. On the contrary, it produces the worst results regarding the AprilTag center-point trajectory. The measurement of the moving-platform orientation appears to be noisy, many spikes can be seen in Fig. 2.4d. Furthermore, there is an error of 1.2 cm at the beginning of the task, which reduces to 0.5 cm at the end. This is because HTC Vive was calibrated at the center of the workspace without rotation about any axis, while the moving-platform trajectory during these experiments does not cross this point. Thus, even though ACROBOT is a small robot, HTC Vive does not provide a high enough measurement accuracy. If it is possible to ensure that the initial pose is very well known, for example by always starting from a physical support that is measured by Creaform C-Track, then estimation of the moving-platform pose is more precise than its measurement by HTC Vive.

When compared to the open-loop control in Section 2.3.1, the results are very similar. Indeed, the image-based method produces a noisy estimation, while the model-based methods are very precise. The only estimation method with some difference in accuracy is the control-based method Est. 1. Indeed, when used with a closed-loop controller it provides the best moving-platform pose estimation accuracy, not surpassing 0.9 cm and 2°. As we now use a closed-loop controller, the controller adapts its output to all perturbations, including the misestimation of the moving-platform pose. For this reason, the control-based moving-platform pose estimation gives better results. Thus, the choice of the moving-platform pose estimation method depends on the choice of the controller.

2.3.3 Approaching a Static Object on ACROBOT by PBVS with Perturbations

This set of experiments has the same initial and desired states, however the initial moving-platform pose is perturbed. That is, at $t = 0\,s$ the assumed moving-platform pose is defined as ${}^{b}\widehat{\mathbf{p}}_{p0} = [0.056\,m, -0.077\,m, 0.308\,m, -6°, -12°, -5°]$, thus the error is 0.072 m and 7°.

The AprilTag trajectory is no longer a straight line, as can be seen in Fig. 2.6a. Note that Est. 2 and Est. 3 appear to be more perturbed than the rest. This can also be seen in the bar graph in Fig. 2.7a. Estimation with the simplest CDPR model has the lowest repeatability of the pixel deviation from the straight line. Despite the issues with repeatability, the

mean and max pixel deviation is lower than for Est. 2 and Est. 3. The rest of model-based estimation methods along with control-based estimation method produce the same mean and maximum pixel deviation. Note that here, as in Fig. 2.5a Est. 5 and Est. 7 have the highest repeatability of both the mean and the maximum pixel deviation.

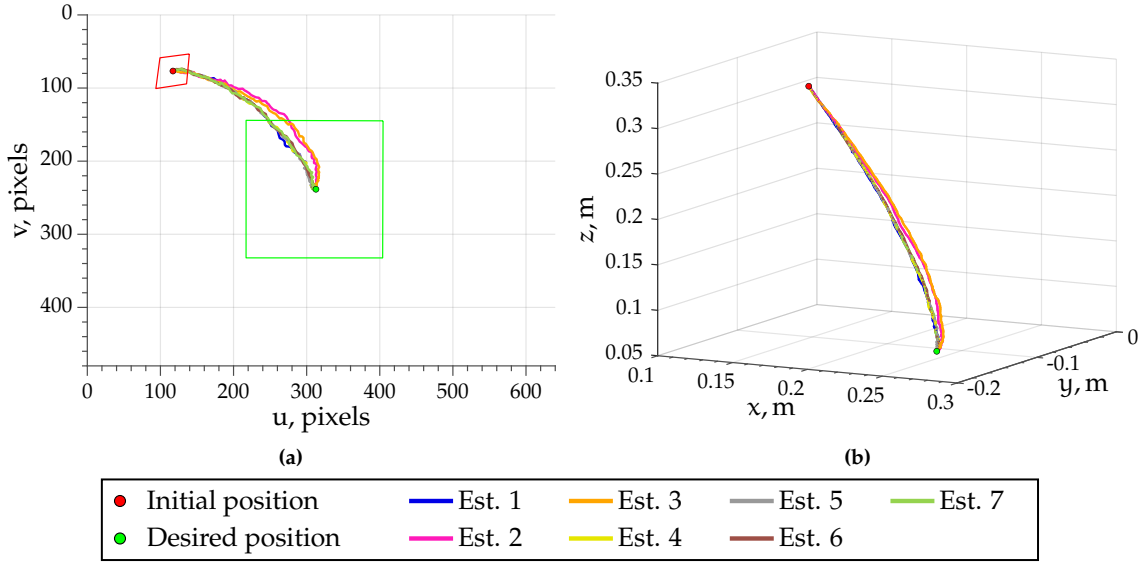


Figure 2.6: Estimation methods on ACROBOT with a static object and perturbations: **(a)** AprilTag center-point trajectory in the image; **(b)** deviation from the ideal straight-line trajectory in the image; **(c)** moving-platform trajectory in the base frame; **(d)** deviation of the moving-platform pose estimation with respect to C-Track measurement

Since the initial moving-platform pose already has an error, it is not surprising that the estimation deviation along the trajectory remains quite large, as can be seen in Fig. 2.7b. However, note that for Est. 1 and Est. 5 through Est. 7 the mean and maximum deviation is very close and the repeatability is almost ideal, the margins being almost invisible on the plot. On the other hand, for Est. 4 the repeatability is about 1 cm and 2°. Interestingly, Est. 2 and Est. 3 show mean values that are even below the initial perturbation. It appears that the perturbation affected the controller with these two estimation methods differently,

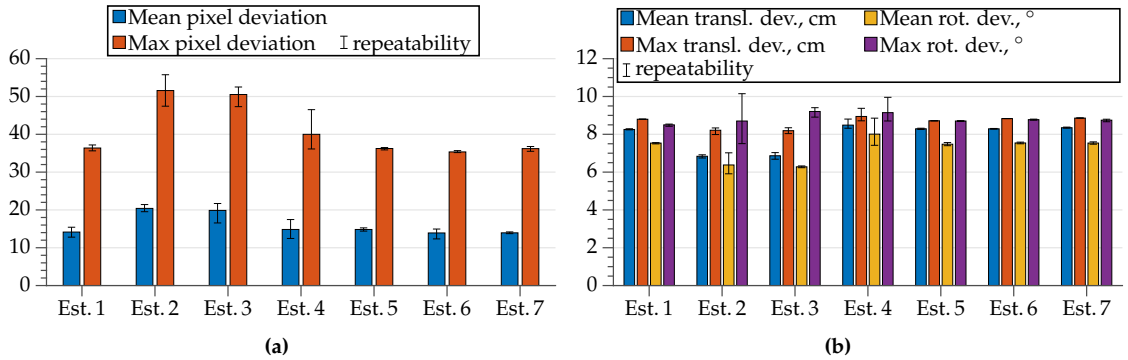


Figure 2.7: Quality of the estimation methods on ACROBOT with perturbation: **(a)** deviation of the AprilTag center-point in the image; **(b)** moving-platform pose estimation deviation with respect to C-Track measurement

the accumulated estimation error actually having the opposite sign compared to the initial perturbation. It is thus surprising to see that these estimation methods produced the worst AprilTag center-point trajectories.

2.3.4 Approaching a Static Object on CAROCA by PBVS

The initial and desired values are the following:

- initial state:
  - $^{b}\mathbf{p}_{p0} = [0.322\,\text{m};\ 0.17\,\text{m};\ 2.018\,\text{m};\ -20°;\ 7°;\ -3°]$
  - $^{c}\mathbf{p}_{o0} = [-0.899\,\text{m};\ 0.775\,\text{m};\ 2.267\,\text{m};\ -157°;\ -7°;\ -179°]$
- desired state:
  - $^{b}\mathbf{p}_{p^*} = [-0.827\,\text{m};\ -1.392\,\text{m};\ 0.89\,\text{m};\ 0°;\ 0°;\ 0°]$
  - $^{c}\mathbf{p}^{*}_{o} = [0\,\text{m};\ 0\,\text{m};\ 0.60\,\text{m};\ 180°;\ 0°;\ 180°]$

As for ACROBOT, the experiments are repeated five times, the first run is shown in Fig. 2.8, while the bar graphs summarizing the results are shown in Fig. 2.9. As the CDPR is significantly larger, so is the trajectory from the initial to the desired state. Furthermore, the cables are steel and their pulleys are large.

The AprilTag center-point trajectory, shown in Fig. 2.8a, is more perturbed than the one shown in Fig. 2.4a. Indeed, even the best result shows a deviation of more than 15 pixels, as can be seen in Fig. 2.8b and 2.9a. Furthermore, here the difference between different estimation methods is more distinct. As in the perturbed case on ACROBOT, here Est. 2 and Est. 3 produce the worst results with almost 35 pixel deviation from the ideal straight-line trajectory. Est. 1 produces a significantly better result, however it is still slightly worse than the model-based approaches. For CAROCA similarly as for ACROBOT, it appears that taking cable elasticity into account does not improve the AprilTag trajectory. On the other hand, taking pulley kinematics into account is beneficial.

Moving-platform trajectory measured by Creaform C-Track is shown in Fig. 2.8c. Note that when using Est. 2 and Est. 3 the produced moving-platform trajectory is significantly different than with other estimation methods. This can be explained by the rapid deviation of the moving-platform pose estimation that can be seen in Fig. 2.8d. Indeed, the translational error rapidly overpasses 0.1 m. The rotational error appears to be small, just over 1°, however at approximately $t = 10\,\text{s}$ this estimation becomes very noisy, which could be affecting the robot behavior. Regarding the other estimation methods, two groups can be distinguished. First, Est. 1 with Est. 4 and Est. 5, where the translational error reaches about 0.07 m. And second, Est. 6 and Est. 7, where the translational error does not surpass 0.04 m. Clearly, when the initial moving-platform pose is well known, using an estimation method that takes pulley kinematics into account leads to the best CDPR behavior.

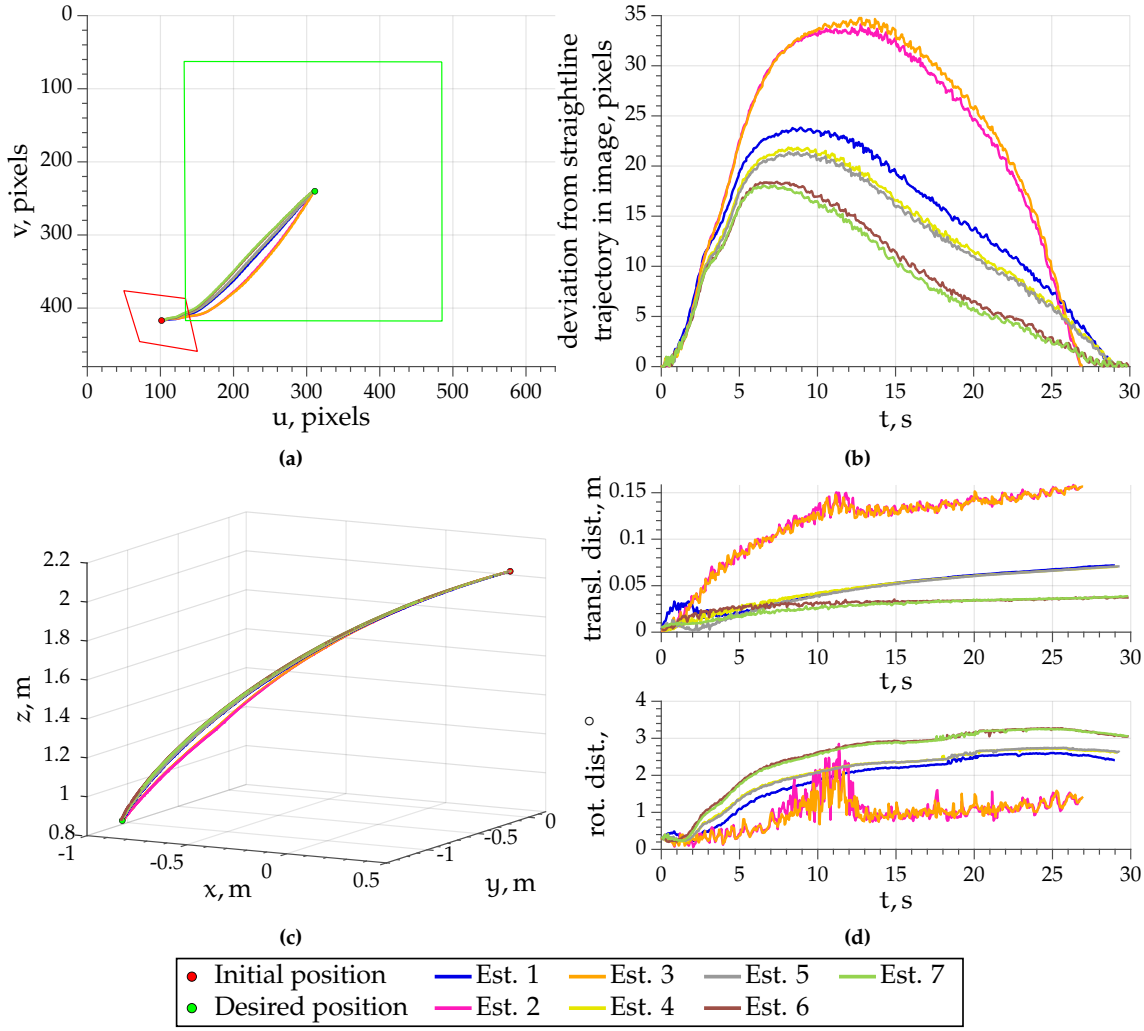


Figure 2.8: Estimation method comparison on CAROCA with a static object: **(a)** AprilTag center-point trajectory in the image; **(b)** deviation from the ideal straight-line trajectory in the image; **(c)** moving-platform trajectory in the base frame; **(d)** deviation of the moving-platform pose estimation with respect to C-Track measurement

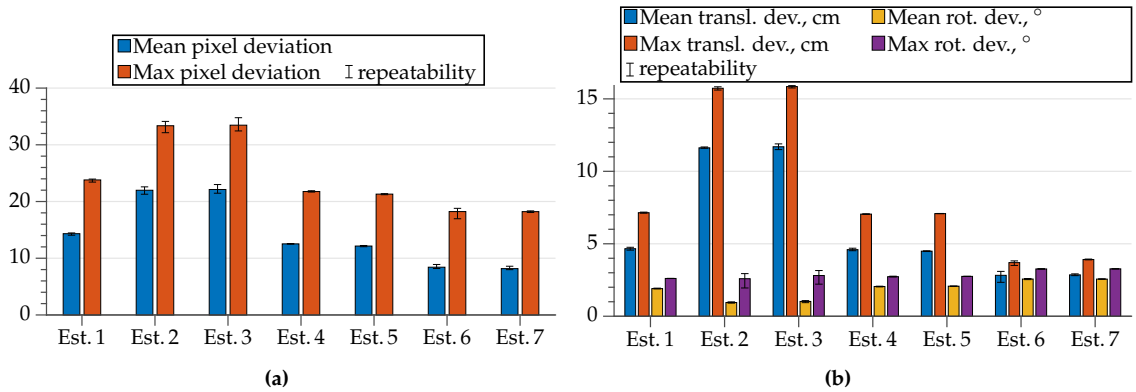


Figure 2.9: Quality of the estimation methods on CAROCA: **(a)** deviation of the AprilTag center-point in the image; **(b)** moving-platform pose estimation deviation with respect to C-Track measurement

2.3.5 Approaching a Static Object on CAROCA by PBVS with Perturbations

The initial and desired state are defined as follows:

- initial state:
  - ◇ ${}^{\mathrm{b}}\mathbf{p}_{\mathrm{p}0} = \left[0.713\,\mathrm{m};\ 0.102\,\mathrm{m};\ 1.899\,\mathrm{m};\ -18°;\ 10°;\ -5°\right]$
  - ◇ ${}^{\mathrm{c}}\mathbf{p}_{\mathrm{o}0} = \left[-1.042\,\mathrm{m};\ 0.709\,\mathrm{m};\ 2.277\,\mathrm{m};\ -159°;\ -10°;\ -179°\right]$
- desired state:
  - ◇ ${}^{\mathrm{b}}\mathbf{p}_{\mathrm{p}^*} = \left[-0.689\,\mathrm{m};\ -1.374\,\mathrm{m};\ 0.757\,\mathrm{m};\ 0°;\ 0°;\ 0°\right]$
  - ◇ ${}^{\mathrm{c}}\mathbf{p}^*_{\mathrm{o}} = \left[0\,\mathrm{m};\ 0\,\mathrm{m};\ 0.6\,\mathrm{m};\ 180°;\ 0°;\ 180°\right]$

The added perturbation is 0.4 m and 8°, making the initial moving-platform pose estimation ${}^{\mathrm{b}}\widehat{\mathbf{p}}_{\mathrm{p}0} = \left[0.561\,\mathrm{m};\ -0.11\,\mathrm{m};\ 2.202\,\mathrm{m};\ -25°;\ 12°;\ 3°\right]$.

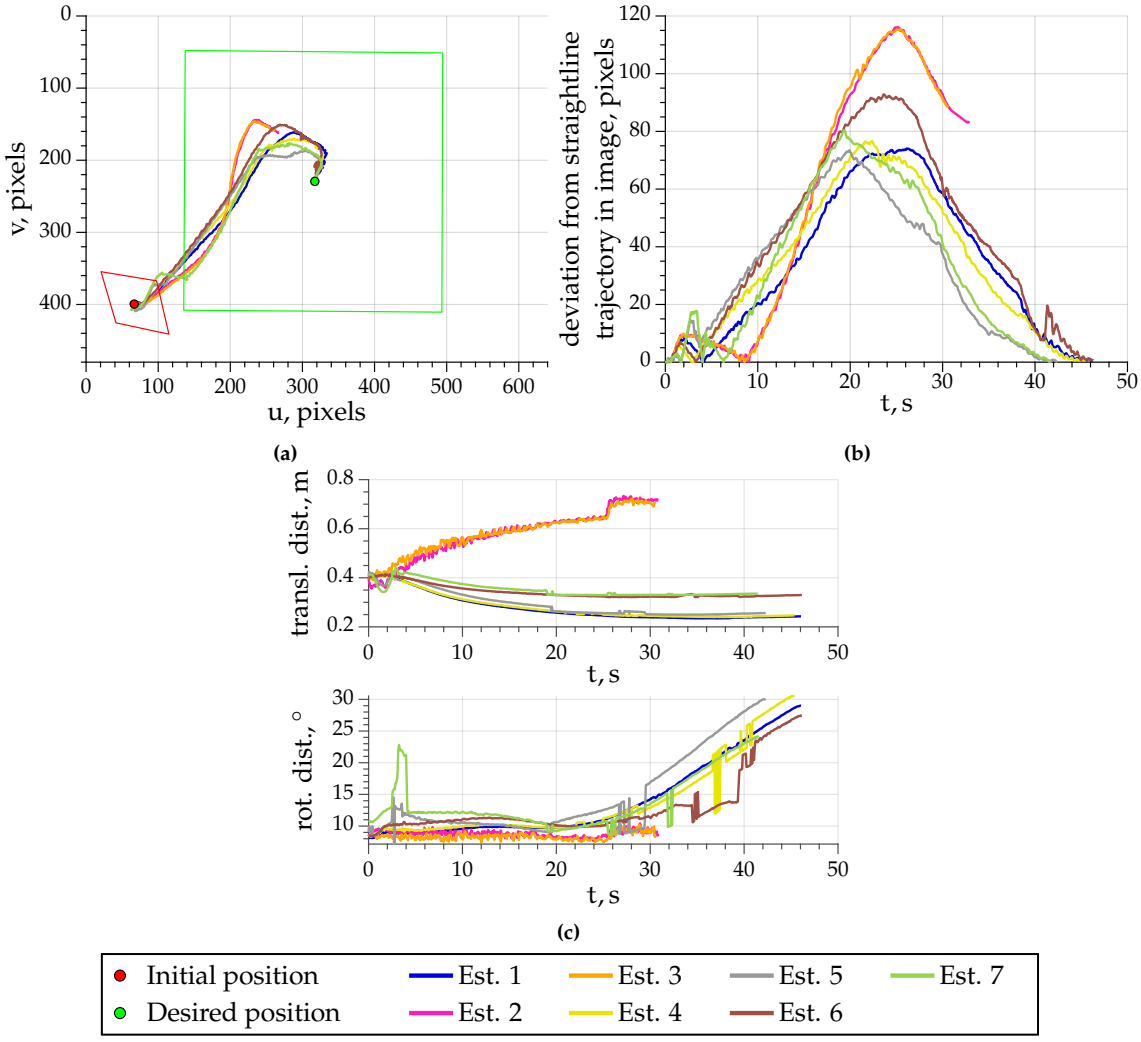


Figure 2.10: Estimation methods on CAROCA with a static object and perturbations: **(a)** AprilTag center-point trajectory in the image; **(b)** deviation from the ideal straight-line trajectory in the image; **(c)** deviation of the moving-platform pose estimation with respect to C-Track measurement

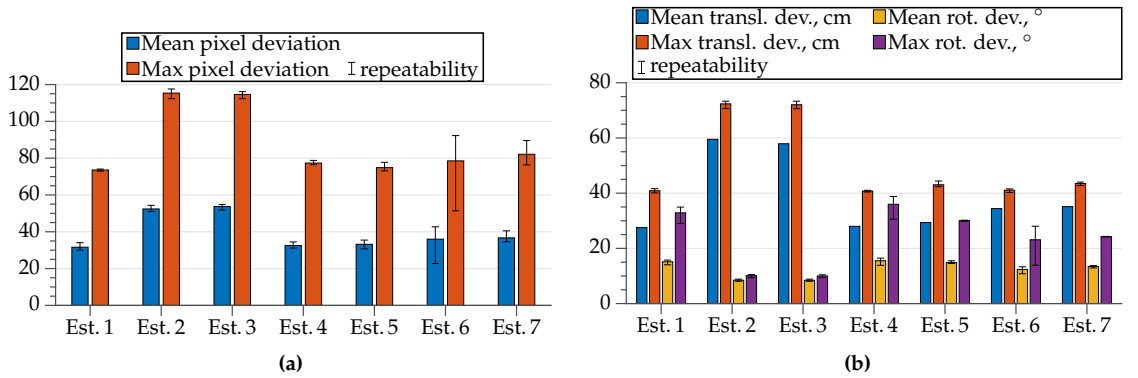


Figure 2.11: Quality of the estimation methods on CAROCA with perturbations: **(a)** deviation of the AprilTag center-point in the image; **(b)** moving-platform pose estimation deviation with respect to C-Track measurement

Having such a large perturbation leads to a large deviation of the AprilTag trajectory from the straight line, as can be seen in Figs. 2.10a and 2.10b. Moreover, when the moving-platform pose is estimated from images, namely in Est. 2 and Est. 3, the task fails. It appears that the same kind of perturbation affects these two estimation methods differently when compared to the other methods. Indeed, here the translational distance only increases, while for Est. 1, Est. 4 and Est. 5 it is almost halved by the end of the task (see Fig. 2.10c). This behavior is consistent among all of the repeated experiments, as can be seen in the bar graphs shown in Fig. 2.11.

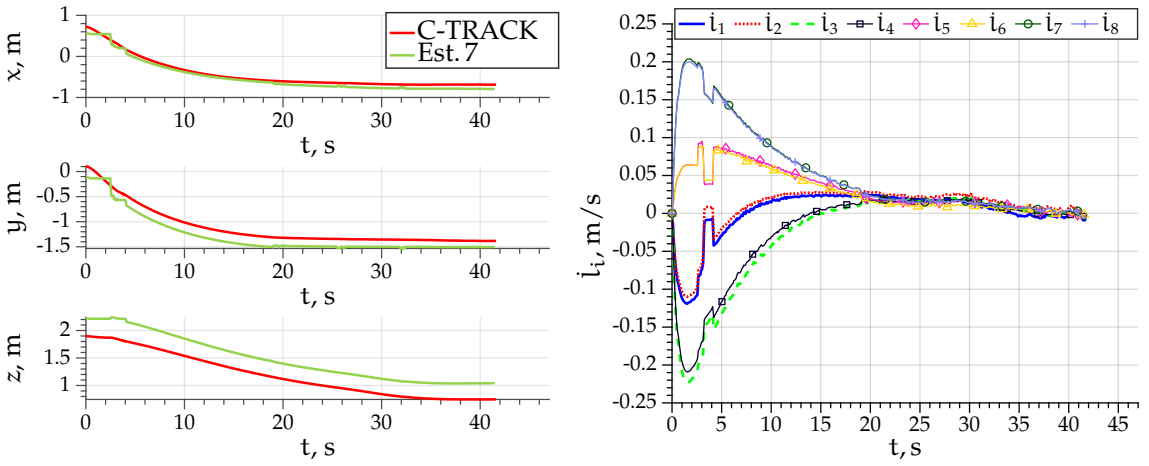


Figure 2.12: Plots for Est. 7: **(a)** translational estimation comparison to C-Track measurement; **(b)** cable velocities

When the system is perturbed, the model-based estimation methods are no longer superior. Indeed, in this case estimation by control integration gives the best results. The results for the model-based estimation methods are very similar, however slightly worse. This can be explained by analyzing Fig. 2.10c. Here for all model-based estimation methods the curves have sudden changes. It can be clearly seen for the rotational difference

between estimation and C-Track measurement, but is also true for the translational distance. As an example, the translational plots of Est. 7 is shown in Fig. 2.12a. Note that the initial difference at $t = 0\,s$ is due to the perturbation that was added to the initial moving-platform pose. In the first three seconds the estimation, shown in green, does not change while C-Track measurement changes. During this time the estimation algorithm was not capable of finding a solution, given the previous moving-platform pose and the new cable lengths. This is not surprising as the initial cable lengths are computed from the initial moving-platform pose, which in this case is erroneous. The new cable lengths are computed as a sum of previous cable lengths and cable length change, where the latter is computed from the motor position change. Once a solution is found, it is a very different pose compared to the previous successful computation. Due to this cable velocities change abruptly at $t = 3\,s$ and then again at $t = 4\,s$ for the same reason. In this case, luckily, once the algorithm was capable of computing a pose, it was indeed in the vicinity of the actual moving-platform pose. However, it is also possible that a pose can be found that significantly increases the distance between the actual and the estimated moving-platform pose and makes the system unstable. Or on the contrary, the algorithm can be unable to compute a new pose for a longer time period, making the distance between the last computed pose and the actual pose large enough to make the system unstable. Thus, model-based estimation methods should only be used when the initial moving-platform pose is known with sufficient accuracy.

2.3.6 Accuracy and Repeatability of the Final Moving-Platform Pose

In this section the goal accuracy and repeatability of the previous experiments are analyzed. Each CDPR repeated the same trajectory altogether 35 times without perturbation and 35 times with perturbation on the initial moving-platform pose. Note that here we do not take into account the experiments on ACROBOT with the moving-platform pose measured by HTC Vive. We also do not take into account the experiments with image-based moving-platform pose estimation in the presence of perturbations for CAROCA, because there the task was not finished successfully and the desired pose was never reached. Thus for CAROCA with perturbations the repeatability is computed out of 25 experiments.

An example of accuracy and repeatability is shown in Fig. 2.13. Here, for a position the repeatability is the radius r_r of the sphere with the origin at $\mathbf{t}_a$ encircling all the position measurements $\mathbf{t}_1$ to $\mathbf{t}_6$. Note that $\mathbf{t}_a$ is the mean of all the position measurements $\mathbf{t}_1$ to $\mathbf{t}_6$. Accuracy is the distance d_a between the mean position $\mathbf{t}_a$ and the desired position $\mathbf{t}_d$.

Thus, we take four sets of 35 moving-platform poses measured by C-Track at the end of each experiment and we find the average pose. Then we find the maximum distance in translation and in rotation from this average pose. The results are shown in Table 2.1. Note that we could not compute the accuracy for CAROCA, because this computation depends on the AprilTag pose measurement with C-Track, which was not possible.

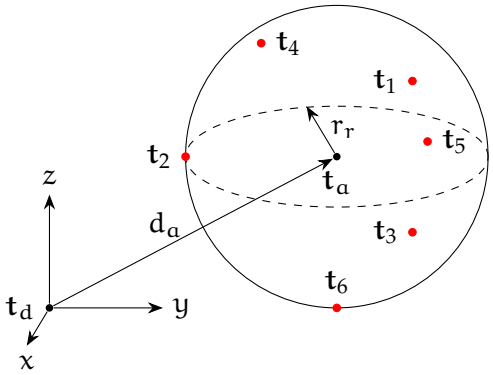


Figure 2.13: Translational parameter $\mathcal{X}$ and its perturbation radius r_x

Table 2.1: Goal repeatability

CDPR	Perturbation	Translation		Rotation	
		Repeatability	Accuracy	Repeatability	Accuracy
ACROBOT	no	0.0010 m	0.0017 m	0.42°	0.68°
ACROBOT	yes	0.0009 m	0.0019 m	0.49°	0.59°
CAROCA	no	0.0015 m	- m	0.17°	-°
CAROCA	yes	0.0016 m	- m	0.31°	-°

In fact, the recorded poses are incredibly close. Even though CAROCA is of considerable size, the repeatability is lower than 2 mm and 0.4°. ACROBOT is of significantly smaller size, thus not surprisingly the translational repeatability is even better here. All of the measurements on ACROBOT are within a sphere with radius of just 1 mm and most of the measurements are within 0.26 mm of the average position. For ACROBOT the rotational repeatability is worse than for CAROCA, while still remaining below 0.5°. A very good accuracy has been shown for ACROBOT: it is below 2 mm and 0.7°.

Finally, as two different controllers were used on ACROBOT to arrive at ${}^{b}\mathbf{p}_{p1}$ from ${}^{b}\mathbf{p}_{p0}$, it is possible to evaluate the accuracy achieved by these controllers. The bar-graph is shown in Fig. 2.14. To evaluate repeatability the experiment with the open-loop velocity controller was repeated 10 times. Similarly, the experiment with the PBVS controller was repeated 30 times. More precisely, five repetitions were done using each of the six moving-platform pose estimation methods in the PBVS control loop. When using PBVS the translational accuracy is five times better, while the rotational accuracy is four times better. Indeed, as the AprilTag is perceived, the task is only finished when the current object pose converges to the desired one and thus a high accuracy can be achieved.

Thus, CDPRs with visual servoing have an excellent accuracy and repeatability.

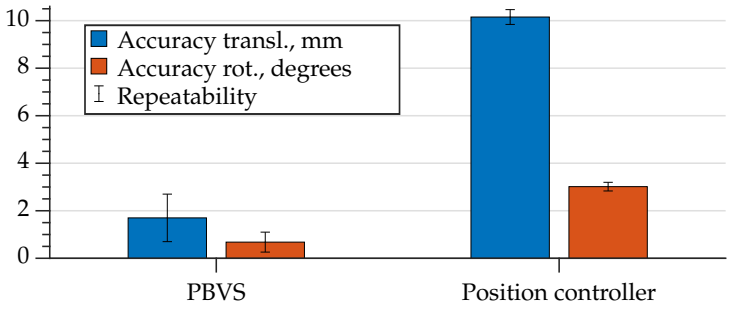


Figure 2.14: ACROBOT moving-platform pose accuracy at the desired pose ${}^{b}\mathbf{p}_{p1}$

2.3.7 Tracking a Mobile Object on ACROBOT

In this subsection ACROBOT is tasked with tracking a small mobile robot that itself is following a black line as shown in Fig. 2.15. With a fully charged battery, each lap takes approximately 16 s. The PBVS controller has no knowledge of the mobile robot trajectory. It simply attempts to keep the AprilTag, which is stuck on the mobile robot, in the desired pose with respect to the camera frame $\mathcal{F}_c$. Since in this task the target is moving, the image-based estimation methods Est. 2 and Est. 3 cannot be used. Each experiment was continued until task failure.

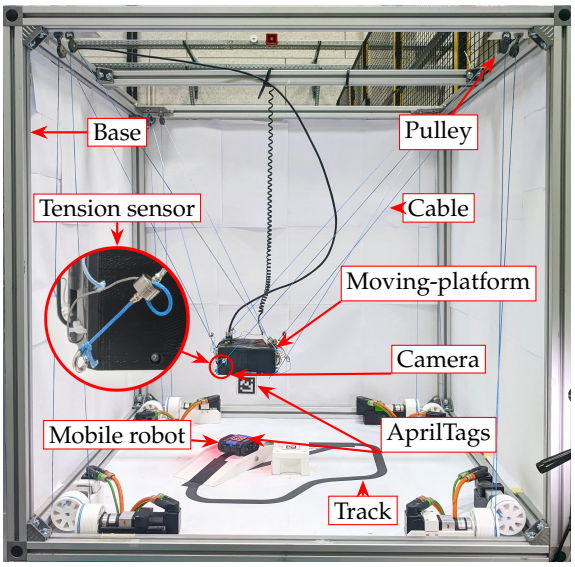


Figure 2.15: ACROBOT with the big moving-platform and tension sensors

The moving-platform trajectory is shown in Figs. 2.16a and 2.16b. The deviation of the moving-platform pose estimation is shown in Fig. 2.16c. Considering the high amount of overlap in Fig. 2.16c, the moving-platform position estimation deviation is shown separately for each estimation method in Fig. 2.17. Note that the axes are equalized for plots in Fig. 2.17 with the exception of Est. 1.

As it can be seen in Fig. 2.17, the use of different moving-platform pose estimation methods leads to a different time of task failure. Indeed, the controller with the estimation

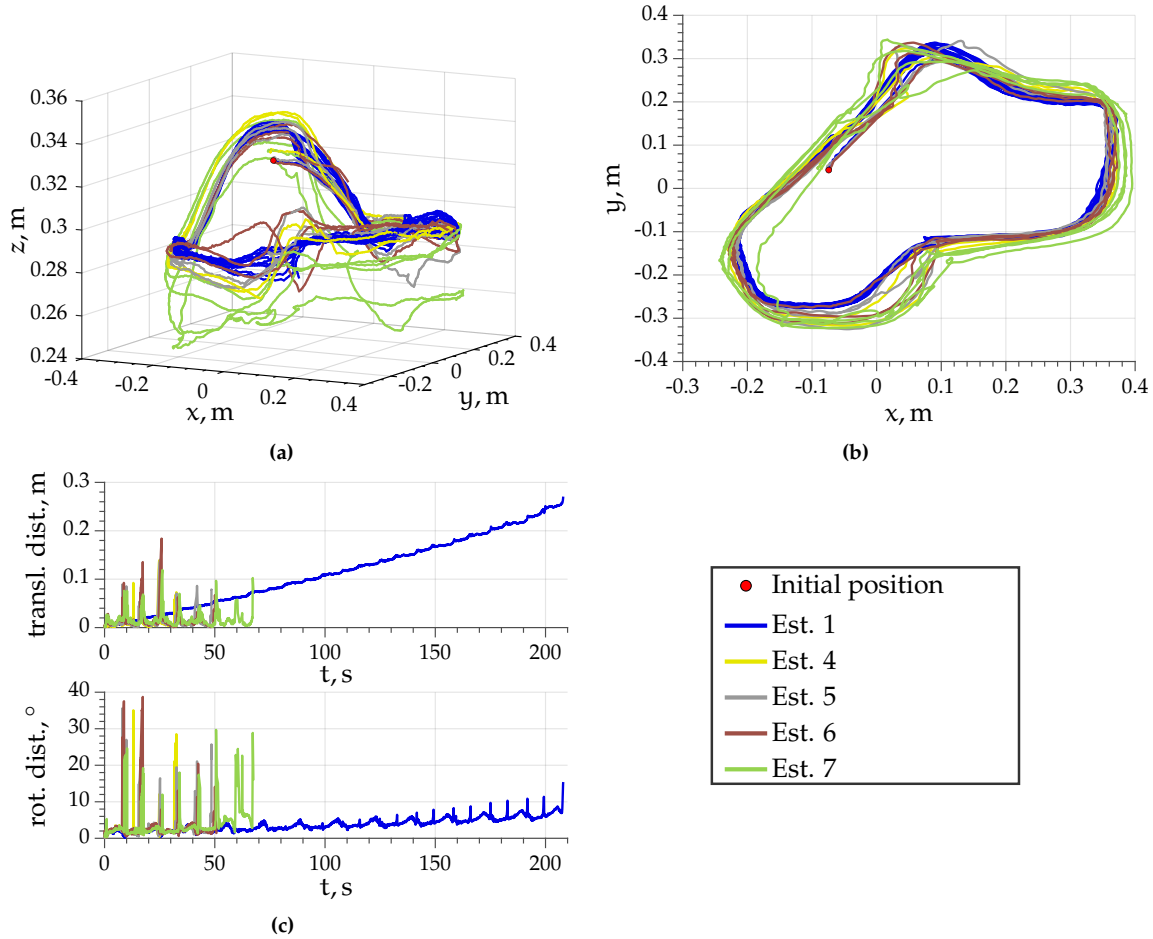


Figure 2.16: Moving-platform trajectory while tracking mobile robot and the estimation deviation from C-Track measurement

method Est. 4, which is based on the simplest CDPR model, is the first one to fail, after only 2 laps (Fig. 2.17a). Experiments with Est. 5 and Est. 6 failed after 3 laps (Figs. 2.17b and 2.17c). When using the estimation method Est. 7 that takes into account cable elasticity and pulley geometry task failure occurs after 4 laps (Fig. 2.17d). Finally, using control integration method Est. 1 the task failure occurs after 12 laps (Fig. 2.17e), which is 3 times more than with Est. 7. It is even more impressive if we take into account the large estimation deviation that occurs. Indeed, the error along z axis is almost 0.3 m at the end. This value far surpasses the estimation error with the model-based methods as can be seen in Fig. 2.16c. On the other hand, the accumulation of the error is smooth, especially for the translational part. This is not the case for the model-based methods, where large and evenly distributed spikes can be seen. Interestingly, all of the model-based method curves have the spikes at the same time. This can also be seen in Figs. 2.17a through 2.17d, where at about $t = 8\,s$, $t = 24\,s$ and $t = 42\,s$ deviation can be seen between the C-Track measurement in blue and the estimation in pink. These problems occur always at the same place on the track. The estimation algorithms take the six cables that are most in tension and estimate the new pose based on these six cables. If there is slack in the system,

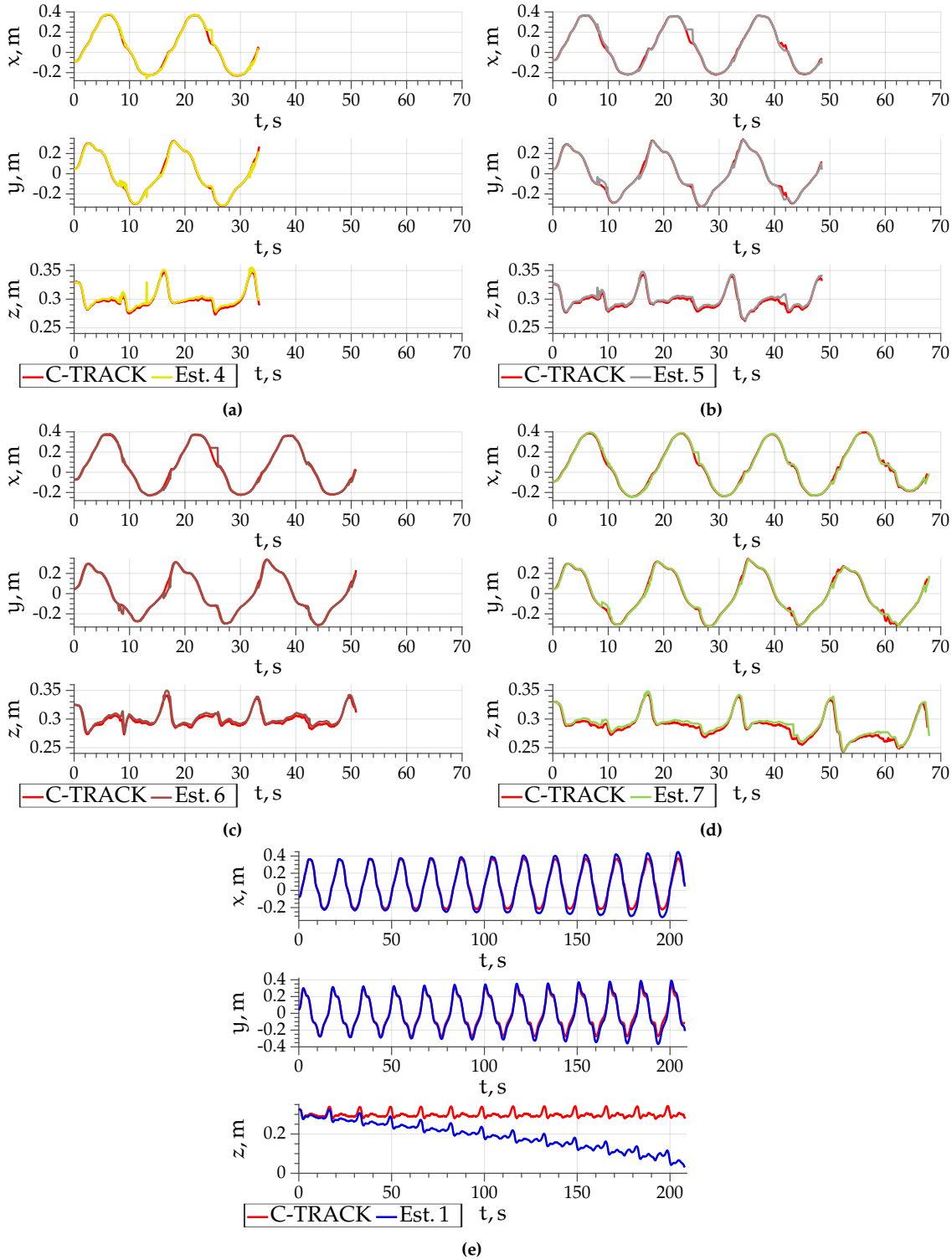


Figure 2.17: The translational deviation of different estimation methods with respect to C-Track measurement

it will have to be transferred between cables as the moving-platform pose changes. With significant slack at this transfer point more than two cables can become slack making the moving-platform underactuated for a short period of time. If this occurs, a slack cable will

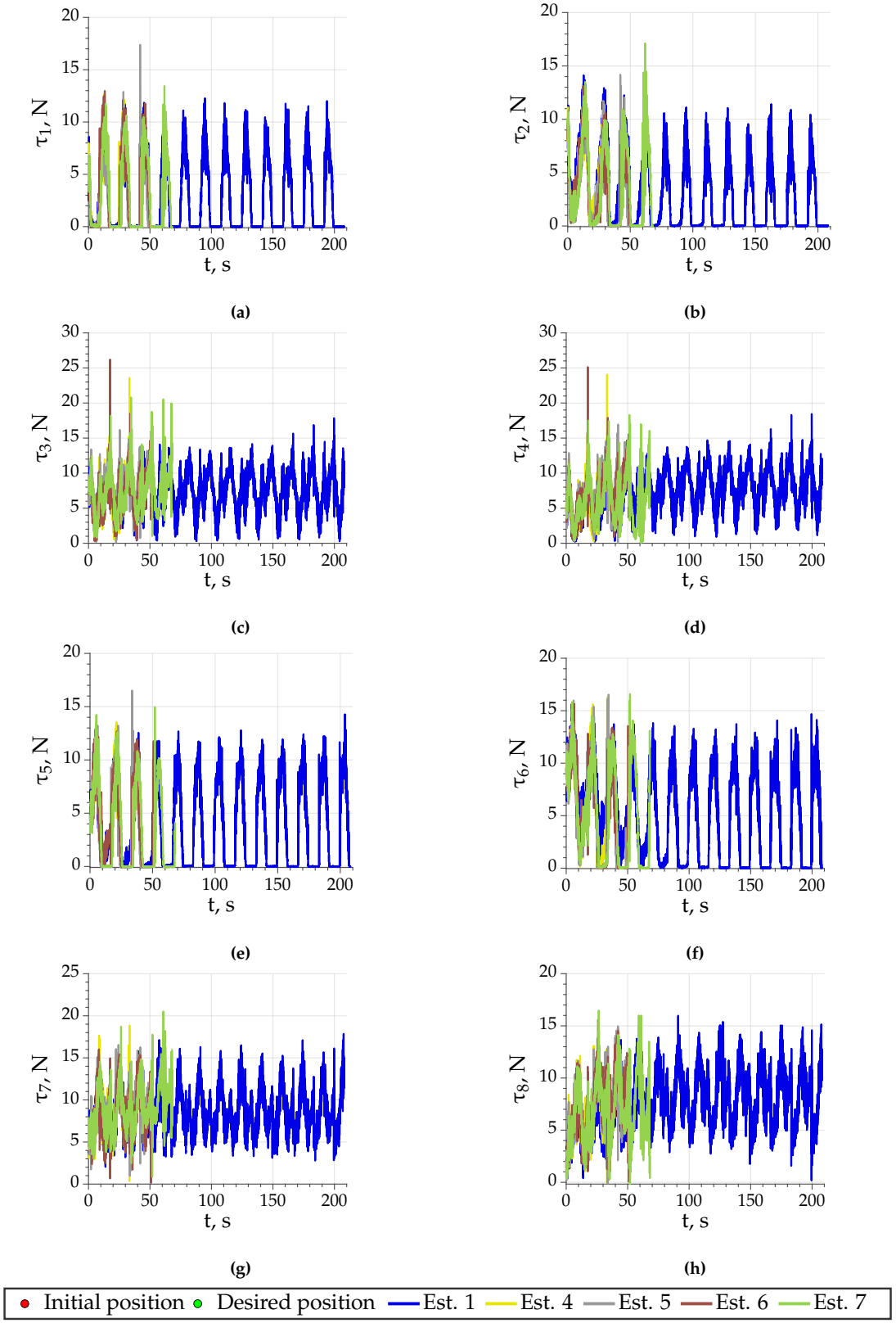


Figure 2.18: Cable tensions τ_i

be used by the estimation algorithm, leading to a wrong moving-platform pose estimation. In fact, at about $t = 24\,s$ the slack is transferred from cables $\mathcal{C}_1$ and $\mathcal{C}_2$ to cables $\mathcal{C}_5$ and $\mathcal{C}_6$ as can be seen in Fig. 2.18. Even though in the next iteration the correct moving-platform pose will probably be found, these spikes of significantly large errors affect the controller in a worse way than a slowly accumulating estimation error as in Est. 1.

2.4 Conclusions

Three moving-platform pose estimation approaches were developed: control-based, image-based and model-based. In the control based approach the control output is integrated to find the new moving-platform pose, given that the previous moving-platform pose is known. In the image-based approach two images are compared and assuming that the initial moving-platform pose is known, the new pose can be computed using frame transformations. Two different image pairs can be used: either two consecutive images or the first and the current image. In the model-based approach moving-platform pose is computed from cable lengths of six cables that are most in tension. Four different models were used: simplest one corresponding to the CDPR model used in control; model taking into account cable elasticity; model taking into account pulley geometry; and model taking into account both cable elasticity and pulley geometry. Thus, altogether seven different methods were tested along with using HTC Vive tracker for online moving-platform pose measurement.

If the initial moving-platform pose is well known, then there will be almost no difference between the seven estimation methods on ACROBOT. Interestingly, moving-platform estimation with image-based approaches is the worst (while still very good), however the produced trajectory is actually slightly better than with the model-based approaches and about as good as for the control-based approach. For CAROCA, the model-based approaches give the best results both in moving-platform pose estimation accuracy and in the produced image trajectory. Furthermore, the models that take into account pulley geometry give the best results. Moving-platform pose estimation with image-based approaches is the worst and the produced trajectory is also the worst. Thus, if it is possible to perfectly know the initial moving-platform pose, then the choice of the pose estimation method will be trivial for ACROBOT, but not for CAROCA. One way of knowing the initial moving-platform pose is using a homing support, on which the moving-platform fits tightly, so that the initial pose is always the same.

When using image-based methods in the presence of perturbations, CAROCA fails its task. With the other methods the task is successfully finished. Furthermore, model-based and control-based approaches produce very similar results. However, it can be seen that for the model-based methods there are abrupt changes in the estimated pose from one iteration to another. In such a case the computed velocities will also change from one instant to another meaning that the CDPR motion can have some sudden jerks.

It is not possible to use image-based methods when the object is in motion. Indeed, it is not possible to discern between the camera motion and the object motion without supplementary observations.

The remaining five estimation methods were tested with a mobile object and the tracking continued until task failure. For the model-based estimation methods, the more precise the model, the longer the time period until task failure. However, even taking into account pulley kinematics and cable elasticity, the tracking continued only for 70 s. On the other hand, using control integration the robot was able to continue tracking the object for 210 s, thus three times longer. The cause to this large difference appears to be cable slackness. Indeed, during the tracking slack accumulates on the cables and at the furthest points of the track the slack rapidly transfers to different cables. At this moment more than two cables are slack and the moving-platform is temporarily underactuated. This leads to using some slack cables in the computation of the moving-platform pose for the model-based methods, which produce an incorrect moving-platform pose estimation. In fact, it appears that computing significantly deviated moving-platform pose for just a few iterations in an overall precise estimation has a worse effect on system stability than slowly and smoothly accumulating estimation error. With model-based estimation methods the moving-platform pose was wrongly estimated at certain points of the track and the task failed rapidly. On the other hand, with control-based estimation the error accumulated smoothly and the task only failed after reaching almost 0.3 m and 15° error.

It should be noted that model-based moving-platform estimation methods can become sensitive to model perturbations, such as badly known cable exit and anchor point coordinates or badly known initial moving-platform pose. This type of perturbation can affect the solvability of the equations. Indeed, even the computation of the initial cable lengths is based on the knowledge of the initial moving-platform pose. Thus it can occur that no solution is found for a given bad previous moving-platform pose estimate and new cable lengths. To conclude, model-based estimation methods should only be used when the initial moving-platform pose is known with sufficient accuracy. Furthermore, when using model-based estimation methods cable slack needs to be avoided.

Thus, control integration is the most robust moving-platform estimation method that can be used for both static and moving targets and that provides a smooth estimation even in the presence of perturbations. Indeed, a smooth rather than mostly precise estimation is the key for the best CDPR behavior.

HTC Vive tracker was used as an online measurement system to directly measure the moving-platform pose. Its accuracy is varying and not certified. The tracker was calibrated in the center of ACROBOT workspace. The initial moving-platform pose during the experiments is not at the center and is indeed measured with a small error. In fact, this leads to having a small perturbation on the initial moving-platform pose, which was avoided for the experiments with moving-platform pose estimation, because

C-Track measurement could be used as the initial knowledge. Overall, if the initial moving-platform pose can be well known, it is preferred to use an estimation method, such as the control-based estimation, instead of measurements by HTC Vive.

3. Stability Analysis and Control Stability Workspace

Contents

3.1 Introduction

In Chapter 2 it was found that PBVS of CDPRs is robust to some perturbations and modeling errors. The simplest CDPR model was used for both CDPRs, namely for ACROBOT and CAROCA presented in Appendix A.1, while the cables of CAROCA are elastic and pulleys have a large radius. Nevertheless, the tasks were successfully completed. When tracking a mobile object and using control-based moving-platform pose estimation method the system appeared to be remarkably robust to errors in the moving-platform pose. Indeed, by the end the accumulated error amounted to 0.25 m and 10°. To quantify the robustness to different perturbations, a stability analysis can be done. It is presented in Section 3.2 and subsequently a novel workspace is defined and computed in Section 3.3.

3.2 Stability Analysis

In visual servoing the Lyapunov stability of the system is usually analyzed. It allows us to determine whether the system is capable of converging to the desired state with the existing uncertainties and modeling errors [Kha02].

From the closed-loop equation (1.40) the stability criterion $\boldsymbol{\Pi}$ of the complete system can be expressed as:

$$\boldsymbol{\Pi} = \mathbf{L}_s \, \mathbf{A}_d^{-1} \mathbf{A}^{\dagger} \widehat{\mathbf{A}} \, \widehat{\mathbf{A}}_d \, \widehat{\mathbf{L}}_s^{\dagger} > \mathbf{0}, \forall t \tag{3.1}$$

It is a sufficient condition, meaning that if $\boldsymbol{\Pi} > \mathbf{0}$, then the system will stable. However if $\boldsymbol{\Pi} \leqslant \mathbf{0}$, then the stability of the system will be unknown.

3.2.1 Stability Analysis of a Simple Planar CDPR with PBVS

Introduction

The choice to first analyze a planar CDPR is due to its simplicity. Indeed, as introduced in Section 1.3.2 having a planar four cable CDPR allows us to have a simplified expression of the forward kinematic Jacobian matrix $\mathbf{A}_{pl}$.

A PBVS is used to control the CDPR, however we must first express its planar version. Thus, in this section, similarly as in Section 1.3.2, all variables are planar: (i) rotation matrices ${}^{i}\mathbf{R}_j$ are (2×2)–matrices; (ii) homogeneous transformation matrices ${}^{i}\mathbf{T}_j$ are (3×3)–matrices; (iii) pose ${}^{i}\mathbf{p}_j$ and velocity ${}^{i}\mathbf{v}_j$ vectors are of size (3×1); (iv) and coordinates ${}^{p}\mathbf{b}_i$ and ${}^{b}\boldsymbol{\alpha}_i$ are of size (2×1).

The feature vector becomes $\mathbf{s}_{pl} = \left[s_x; s_y; \theta_p\right]^{\top}$. Thus the error is $\boldsymbol{e}_{pl} = \mathbf{s}_{pl} - \mathbf{s}_{pl}^{*}$, where $\mathbf{s}_{pl}^{*}$ is the desired feature vector.

Interaction matrix $\mathbf{L_{s,pl}}$ that was defined for PBVS in (1.47) simplifies to:

$$\mathbf{L_{s,pl}} = \begin{bmatrix} -1 & 0 & e_y \\ 0 & -1 & -e_x \\ 0 & 0 & -1 \end{bmatrix} \tag{3.2}$$

where e_x and e_y are the first two components of $\boldsymbol{e}_{pl}$. Note that the planar interaction matrix does not depend on the rotational error, but only on the two components of the translational error.

Thus we can express $\dot{\boldsymbol{e}}_{pl}$ as:

$$\dot{\boldsymbol{e}}_{pl} = \mathbf{L_{s,pl}}{}^{c}\mathbf{v}_c \tag{3.3}$$

The system model defined in (1.38) can be written for the planar CDPR in the following form:

$$\dot{\boldsymbol{e}}_{pl} = \mathbf{L_{s,pl}}\,\mathbf{A}_{d,pl}^{-1}\,\mathbf{A}_{pl}^{\dagger}\,\dot{\mathbf{l}}_{pl} \tag{3.4}$$

and the cable velocity vector $\dot{\mathbf{l}}_{pl}$ can be expressed as:

$$\dot{\mathbf{l}}_{pl} = -\lambda\,\widehat{\mathbf{A}}_{pl}\,\widehat{\mathbf{A}}_{d,pl}\,\widehat{\mathbf{L}}_{\mathbf{s,pl}}^{-1}\,\boldsymbol{e}_{pl} \tag{3.5}$$

where the inverse of the interaction matrix is computed as:

$$\widehat{\mathbf{L}}_{\mathbf{s,pl}}^{-1} = \begin{bmatrix} -1 & 0 & -\hat{e}_y \\ 0 & -1 & \hat{e}_x \\ 0 & 0 & -1 \end{bmatrix} \tag{3.6}$$

and the Adjoint matrix (1.37) for the planar CDPR simplifies to [Ead14]:

$$\mathbf{A}_{d,pl} = \begin{bmatrix} {}^{p}\mathbf{R}_c & \mathbf{E}^{\top}{}^{p}\mathbf{t}_c \\ 0\ \ 0 & 1 \end{bmatrix} \tag{3.7}$$

with ${}^{p}\mathbf{t}_c$ being the (2×1)–vector pointing from O_p to O_c; and $\mathbf{E} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$.

Thus, the stability criterion $\boldsymbol{\Pi}$ to ensure the global asymptotic stability (GAS) is expressed for the planar case as:

$$\boldsymbol{\Pi}_{pl} = \mathbf{L_{s,pl}}\,\mathbf{A}_{d,pl}^{-1}\,\mathbf{A}_{pl}^{\dagger}\,\widehat{\mathbf{A}}_{pl}\,\widehat{\mathbf{A}}_{d,pl}\,\widehat{\mathbf{L}}_{\mathbf{s,pl}}^{-1} \tag{3.8}$$

Accordingly, if the estimations of the Jacobian matrix $\widehat{\mathbf{A}}_{pl}$, the adjoint matrix $\widehat{\mathbf{A}}_{d,pl}$ and the interaction matrix $\widehat{\mathbf{L}}_{\mathbf{s,pl}}$ are not too coarse, $\mathbf{\Pi}_{pl}$ will be near the identity matrix, and hence positive definite, ensuring that the system is GAS.

Here, the stability analysis is performed in three steps with increasing difficulty. First, only perturbations in the vision system, i.e., perturbations in object pose estimation, are taken into account (*Stability Analysis I*). Then, errors in the pose of the camera w.r.t. $\mathcal{F}_p$ are considered (*Stability Analysis II*). Finally, the CDPR model errors are examined (*Stability Analysis III*).

Stability Analysis I

While considering the errors in the vision system only, matrices $\widehat{\mathbf{A}}_{pl}$ and $\widehat{\mathbf{A}}_{d,pl}$ are assumed to be determined accurately. Thus, the stability criterion (3.8) is simplified to:

$$\mathbf{\Pi}_{p1} = \mathbf{L}_{\mathbf{s,pl}}\widehat{\mathbf{L}}_{\mathbf{s,pl}}^{-1} = \begin{bmatrix} 1 & 0 & \hat{e}_y - e_y \\ 0 & 1 & e_x - \hat{e}_x \\ 0 & 0 & 1 \end{bmatrix} > 0 \tag{3.9}$$

The symmetric part of $\mathbf{\Pi}_{p1}$, named $\left(\mathbf{\Pi}_{p1}\right)_{Sym}$, is obtained as follows:

$$\left(\mathbf{\Pi}_{p1}\right)_{Sym} = \frac{1}{2}\left(\mathbf{L}_{\mathbf{s,pl}}\widehat{\mathbf{L}}_{\mathbf{s,pl}}^{-1} + \left(\mathbf{L}_{\mathbf{s,pl}}\widehat{\mathbf{L}}_{\mathbf{s,pl}}^{-1}\right)^{\top}\right) = \begin{bmatrix} 1 & 0 & \frac{\hat{e}_y - e_y}{2} \\ 0 & 1 & \frac{e_x - \hat{e}_x}{2} \\ \frac{\hat{e}_y - e_y}{2} & \frac{e_x - \hat{e}_x}{2} & 1 \end{bmatrix} \tag{3.10}$$

Consequently, the eigenvalues of $\left(\mathbf{\Pi}_{p1}\right)_{Sym}$ are:

$$\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{(e_x - \hat{e}_x)^2 + (e_y - \hat{e}_y)^2}}{2} + 1 \\ -\frac{\sqrt{(e_x - \hat{e}_x)^2 + (e_y - \hat{e}_y)^2}}{2} + 1 \\ 1 \end{bmatrix} \tag{3.11}$$

For (3.9) to hold, the eigenvalues λ_1, λ_2, and λ_3 must all be positive [Joh70]. Note that λ_1 is always positive, because the term under the square root is always non-negative. Since $\mathbf{e}_{pl} = \mathbf{s}_{pl} - \mathbf{s}^*_{pl}$ and $\hat{\mathbf{s}}^*_{pl} = \mathbf{s}^*_{pl}$, then λ_2 will be positive if and only if the following condition holds true:

$$\sqrt{\Delta s_x^2 + \Delta s_y^2} < 2 \tag{3.12}$$

where $\Delta s_x = s_x - \hat{s}_x$ and $\Delta s_y = s_y - \hat{s}_y$.

Thus, we can conclude that the distance $\Delta s_{pl} = \sqrt{\Delta s_x^2 + \Delta s_y^2}$ between the current object pose $\mathbf{s}_{pl}$ and its estimation $\hat{\mathbf{s}}_{pl}$ must be smaller than 2 meters. This of course is only true if there are indeed no other perturbations in the system.

It is important to understand the control behavior when the estimation $\hat{\mathbf{s}}_{pl}$ is not sufficiently close to $\mathbf{s}_{pl}$. If a bias[3] exists on $\mathbf{s}_{pl}$, but not on $\mathbf{s}^*_{pl}$, then the desired state will never be reached. More precisely, the error $\boldsymbol{e}_{pl}$ will converge to zero, while the actual desired state will not be reached. Thus, it can be concluded, that a bias on $\mathbf{s}_{pl}$, but not on $\mathbf{s}^*_{pl}$ leads to a bad accuracy. To avoid this, $\mathbf{s}^*_{pl}$ needs to be measured with the same bias as $\mathbf{s}_{pl}$. In this case, as the error $\boldsymbol{e}_{pl}$ converges to zero, the system will converge to $\mathbf{s}^*_{pl}$ and will be stable. This conclusion is also true for the spatial case, discussed later in the manuscript.

Note that this is true for any CDPR because no perturbation in the CDPR model parameters is considered here. More precisely, this is true for any robot, because here only the pure visual servoing part of the control is considered.

Stability Analysis II

Here, the hand-eye calibration errors are also taken into account. That is, the errors in the camera pose in $\mathcal{F}_p$ are considered along with the ones described in *Stability Analysis I*. Accordingly, matrix $\mathbf{A}_{d,pl}$, expressed in (3.7), is considered in the stability criterion (3.8), which becomes:

$$\boldsymbol{\Pi}_{p2} = \mathbf{L}_{\mathbf{s,pl}}\,\mathbf{A}_{d,pl}^{-1}\widehat{\mathbf{A}}_{d,pl}\,\widehat{\mathbf{L}}_{\mathbf{s,pl}}^{-1} = \begin{bmatrix} & & d_1 \\ & \mathbf{R}_\psi & \\ & & d_2 \\ 0 & 0 & 1 \end{bmatrix} > 0 \tag{3.13}$$

where:

$$\mathbf{R}_\psi = {}^c\mathbf{R}_p\,{}^p\widehat{\mathbf{R}}_c = \begin{bmatrix} \cos(\psi) & -\sin(\psi) \\ \sin(\psi) & \cos(\psi) \end{bmatrix} \tag{3.14}$$

$$d_1 = \hat{e}_y\cos(\psi) - e_y + \hat{e}_x\sin(\psi) - \Delta t_y\cos(\theta_{pc}) + \Delta t_x\sin(\theta_{pc}) \tag{3.15}$$

$$d_2 = -\hat{e}_x\cos(\psi) + e_x + \hat{e}_y\sin(\psi) + \Delta t_y\sin(\theta_{pc}) + \Delta t_x\cos(\theta_{pc}) \tag{3.16}$$

and ψ is the rotational error between θ_{pc} and its estimation $\widehat{\theta}_{pc}$.

Note that the camera pose in the moving-platform frame $\mathcal{F}_p$ is expressed as ${}^p\mathbf{p}_c = \begin{bmatrix} {}^p\mathbf{t}_c{}^\top & \theta_{pc} \end{bmatrix}^\top = \begin{bmatrix} t_x & t_y & \theta_{pc} \end{bmatrix}^\top$. Furthermore, $\Delta t_x = t_x - \widehat{t}_x$ and $\Delta t_y = t_y - \widehat{t}_y$.

[3] A bias is a systematic measurement error with a non-zero mean.

The eigenvalues of $\left(\mathbf{\Pi}_{p2}\right)_{Sym}$ are the following:

$$\begin{bmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\end{bmatrix} = \begin{bmatrix}\cos(\psi)\\ \frac{\cos(\psi)+1+\sqrt{(1-\cos(\psi))^2+d_1^2+d_2^2}}{2}\\ \frac{\cos(\psi)+1-\sqrt{(1-\cos(\psi))^2+d_1^2+d_2^2}}{2}\end{bmatrix} \tag{3.17}$$

λ_1 is positive if and only if

$$|\psi| = |\theta_{pc} - \hat{\theta}_{pc}| < \pi/2 \tag{3.18}$$

λ_2 is always positive, because the term under the square root is non-negative, given that (3.18) is held. Finally, due to the complexity of the expression, λ_3 is analyzed numerically.

By analyzing the stability criterion $\mathbf{\Pi}_{p2}$ and the eigenvalues of $\left(\mathbf{\Pi}_{p2}\right)_{Sym}$, it is clear that the camera position in the moving-platform frame does not have any effect on system stability. Indeed, it is the difference between the actual position ${}^{p}\mathbf{t}_c$ and its estimation ${}^{p}\hat{\mathbf{t}}_c$ that can have an effect on the system stability, but not those values themselves. This is not true for the other variables, i.e. $\boldsymbol{e}_t$ and θ_{pc} and their estimations.

In the numerical analysis it was found that there is no single answer. Indeed, an infinite amount of solutions is possible, because stability criterion is affected by multiple parameters at the same time. Furthermore, these parameters are inter-dependent. More precisely, by reducing value of one parameter, it becomes possible to increase the value of another without making the system unstable. Some illustrative examples are shown in Fig. 3.1, where each vertical line corresponds to one of the possible combinations. For instance, from Combination (Cb.) 2 the system will be stable if:

$$\begin{cases} |\boldsymbol{e}_{pl}| = \begin{cases} |e_x| = |s_x - s_x^*| \leqslant 5.0\,\mathrm{m} \\ |e_y| = |s_y - s_y^*| \leqslant 5.0\,\mathrm{m} \end{cases} \\ |\Delta\boldsymbol{e}_{pl}| = \begin{cases} |\Delta e_x| = |e_x - \hat{e}_x| \leqslant 0.5\,\mathrm{m} \\ |\Delta e_y| = |e_y - \hat{e}_y| \leqslant 0.5\,\mathrm{m} \end{cases} \\ |\Delta\mathbf{t}| = \begin{cases} |\Delta t_x| \leqslant 0.3\,\mathrm{m} \\ |\Delta t_y| \leqslant 0.3\,\mathrm{m} \end{cases} \\ |\Delta\theta| = |\psi| \leqslant 6^\circ \end{cases}$$

From Cb. 13, the system will be stable if:

$$\begin{cases} |\boldsymbol{e}_{pl}| = \begin{cases} |e_x| = |s_x - s_x^*| \leqslant 0.5\,\mathrm{m} \\ |e_y| = |s_y - s_y^*| \leqslant 0.5\,\mathrm{m} \end{cases} \\ |\Delta \boldsymbol{e}_{pl}| = \begin{cases} |\Delta e_x| = |e_x - \hat{e}_x| \leqslant 0.3\,\mathrm{m} \\ |\Delta e_y| = |e_y - \hat{e}_y| \leqslant 0.3\,\mathrm{m} \end{cases} \\ |\Delta \mathbf{t}| = \begin{cases} |\Delta t_x| \leqslant 0.2\,\mathrm{m} \\ |\Delta t_y| \leqslant 0.2\,\mathrm{m} \end{cases} \\ |\Delta\theta| = |\psi| \leqslant 61^\circ \end{cases}$$

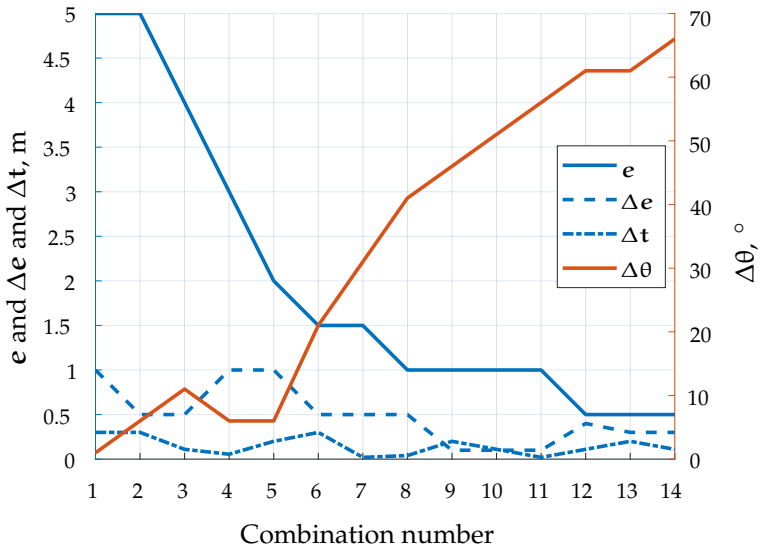


Figure 3.1: Example of possible combinations of design variables ($\boldsymbol{e}$, $\Delta\boldsymbol{e}$, $\Delta\mathbf{t}$, $\Delta\theta$) for a stable system

It should be noted, that for any combination, each variable can take any value within the chosen limit. For example, in Cb. 2 detailed above the rotational error $\Delta\theta$ can be of any value from -6° to 6°, while in Cb. 13 it can be any value from -61° to 61°. Therefore, it can be concluded that the system is exhibiting very strong robustness.

It should also be noted that at this level the stability analysis is still independent of the robot model. Indeed, only the pure visual servoing parameters and the hand-eye calibration in the form of camera pose in $\mathcal{F}_p$ are analyzed here.

Stability Analysis III

Finally, the whole system is analyzed, including perturbations on the robot model: the moving-platform pose w.r.t. $\mathcal{F}_b$, the exit points A_i and anchor points B_i. This means that this stability analysis requires us to define the robot model. Since no planar CDPR was available for this thesis, we use a planar version of ACROBOT prototype for this analysis (see Appendix A.1.1). In short: (i) the WS size of the robot is supposed to be equal to $1\,\mathrm{m}\times 1\,\mathrm{m}$; (ii) the origin of $\mathcal{F}_b$ is located at the WS center; (iii) the moving-platform is

assumed to be 0.11 m×0.11 m, thus the maximum offset between the origins of $\mathcal{F}_p$ and $\mathcal{F}_c$ is equal to 0.055 m.

Here, the stability criterion $\mathbf{\Pi}_{pl}$ cannot be simplified and keeps the form defined in (3.8). Due to its complexity, especially the need for the pseudo-inverse of the Jacobian matrix $\mathbf{A}_{pl}^{\dagger}$, the stability analysis is performed numerically.

As in *Stability analysis II*, once again it is not possible to arrive at one single solution, because of the many interacting parameters and perturbations. Some sets of variable ranges that ensure system stability are shown in Fig. 3.2.

Some other characteristics were also observed:

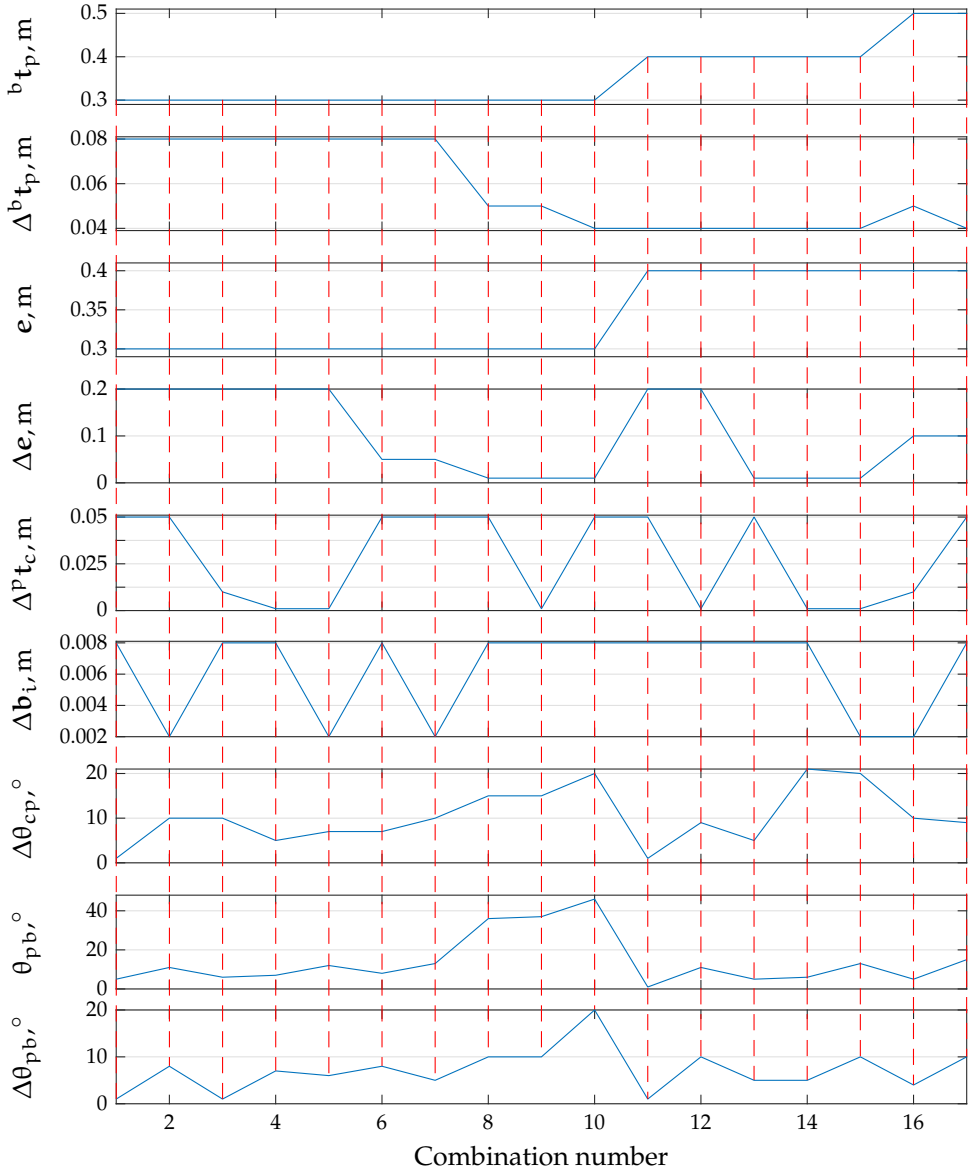


Figure 3.2: Example of possible combinations of design variables (${}^{b}\mathbf{t}_p$, $\Delta^{b}\mathbf{t}_p$, $\mathbf{e}$, $\Delta\mathbf{e}$, $\Delta^{p}\mathbf{t}_c$, $\Delta^{p}\mathbf{b}_i$, $\Delta\theta_{pc}$, θ_{pb}, $\Delta\theta_{pb}$) for a stable system

i. The smaller the desired displacement range of the moving-platform w.r.t the WS center, the larger the tolerated perturbation $\Delta^{b}\mathbf{t}_{p}$.
ii. The difference between $\boldsymbol{e}$ and $\hat{\boldsymbol{e}}$ must always be smaller than $\boldsymbol{e}$ itself.
iii. ${}^{p}\mathbf{t}_{c}$ and θ_{pc} range do not have any effect on stability. Indeed, the camera can be put anywhere on the moving-platform;
iv. Reducing $\Delta^{p}\mathbf{t}_{c}$ often leads to significant increase in the rotational range for θ_{pb}.
v. Similarly, reducing $\Delta\mathbf{b}_{i}$ allows having larger rotational errors of the moving-platform.
vi. If $\Delta^{b}\mathbf{t}_{p}$ and $\Delta\boldsymbol{e}_{pl}$ are comparatively small, and the moving-platform is not in the vicinity of the WS limits, platform could rotate up to 46°. The closer the moving-platform to the center of the WS, the larger its range of rotation.

In conclusion, despite having so many input variables, the system is highly robust to a wide range of perturbations. Furthermore, considering the first characteristic from the list above, it appears that the stability criterion $\boldsymbol{\Pi}$ depends on the moving-platform pose. This case study is continued in Section 3.3.2.

3.2.2 Stability Analysis of a Spatial CDPR with PBVS

As shown in the theoretical analysis of a planar CDPR in Section 3.2.1, doing stability analysis of a closed-loop system allows us to evaluate its robustness to different perturbations.

In this section, stability analysis is divided into three subsections depending on the perturbations that are being taken into account.

Stability Analysis I

When considering only the errors in the vision system, the stability criterion (3.1) simplifies to:

$$\boldsymbol{\Pi}_{s1} = \mathbf{L}_{\mathbf{s}}\widehat{\mathbf{L}}_{\mathbf{s}}^{-1} = \begin{bmatrix} \mathbf{I}_3 & -\Big([\boldsymbol{e}_t]_\times - [\hat{\boldsymbol{e}}_t]_\times\Big)\widehat{\mathbf{L}}_{\omega}^{-1} \\ \mathbf{0}_3 & \mathbf{L}_{\omega}\widehat{\mathbf{L}}_{\omega}^{-1} \end{bmatrix} = \begin{bmatrix} \mathbf{I}_3 & -[\Delta\boldsymbol{e}_t]_\times\widehat{\mathbf{L}}_{\omega}^{-1} \\ \mathbf{0}_3 & \mathbf{L}_{\omega}\widehat{\mathbf{L}}_{\omega}^{-1} \end{bmatrix} > 0 \tag{3.19}$$

where $\Delta\boldsymbol{e}_t = \boldsymbol{e}_t - \hat{\boldsymbol{e}}_t = \left(\mathbf{s}_t - \mathbf{s}_t^*\right) - \left(\hat{\mathbf{s}}_t + \mathbf{s}_t^*\right) = \mathbf{s}_t - \hat{\mathbf{s}}_t$.

Upon examination of $\boldsymbol{\Pi}_{s1}$ it is clear that the translational error $\boldsymbol{e}_t$ or indeed object position $\mathbf{s}_t$ (or the desired one $\mathbf{s}_t^*$) has no effect on the stability criterion. In fact, it is the difference between the current position $\mathbf{s}_t$ and its estimation $\hat{\mathbf{s}}_t$ that can affect the stability criterion. Interestingly, it is not the same with rotations. The upper right term $-[\Delta\boldsymbol{e}_t]_\times\widehat{\mathbf{L}}_{\omega}^{-1}$ contains $\widehat{\mathbf{L}}_{\omega}^{-1}$, where the estimation of $\theta\mathbf{u}$ is present. Thus, not only the difference between the current object orientation and its estimation, but also the values themselves can affect the stability criterion.

Since $\boldsymbol{\Pi}_{s1}$ is a (6×6)–matrix, the analytical stability analysis of this criterion turns out to be very complex. The numerical results are shown in Fig. 3.3.

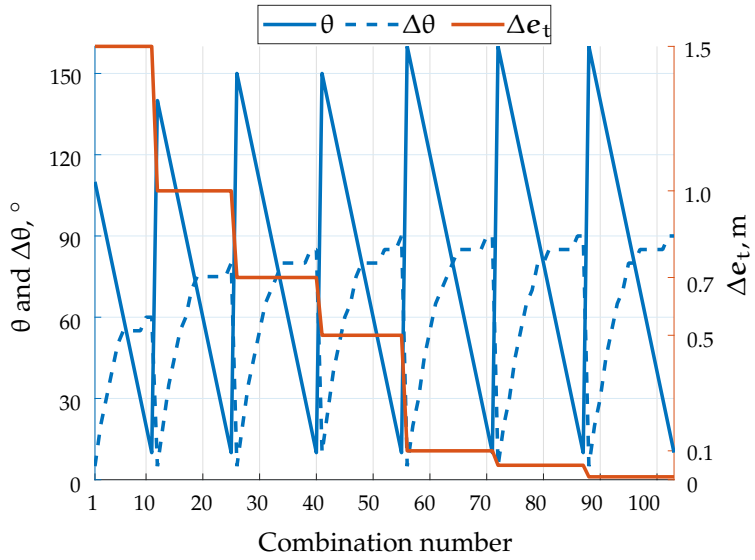


Figure 3.3: Example of possible combinations of design variables ($\Delta\boldsymbol{e}$, θ, $\Delta\theta$)

Once again the variable ranges are interdependent. It means that the lower $\Delta\boldsymbol{e}$, the higher the acceptable range for θ and/or $\Delta\theta$. The following maximum values were found: (i) $\Delta\boldsymbol{e}$ can be at most 1.5 m (Fig. 3.3 Cb. 1 through 10); (ii) θ can be at most 160° (e.g. Cb. 56); (iii) $\Delta\theta$ can be at most 90° (e.g. Cb. 55).

Stability Analysis II

Here, only the Jacobian matrix is assumed to be perfectly known. Therefore, the stability criterion (3.1) becomes:

$$\boldsymbol{\Pi}_{s2} = \mathbf{L}_s \, \mathbf{A}_d^{-1} \widehat{\mathbf{A}}_d \, \widehat{\mathbf{L}}_s^{-1} = \begin{bmatrix} {}^c\mathbf{R}_p \, {}^p\widehat{\mathbf{R}}_c & \mathbf{d}_3 \\ \mathbf{0}_3 & \mathbf{L}_\omega \, {}^c\mathbf{R}_p \, {}^p\widehat{\mathbf{R}}_c \, \widehat{\mathbf{L}}_\omega^{-1} \end{bmatrix} > 0 \tag{3.20}$$

where $\mathbf{d}_3 = \left({}^c\mathbf{R}_p \, {}^p\widehat{\mathbf{R}}_c [\hat{\boldsymbol{e}}_t]_\times - [\boldsymbol{e}_t]_\times \, {}^c\mathbf{R}_p \, {}^p\widehat{\mathbf{R}}_c - {}^c\mathbf{R}_p [\Delta^p\mathbf{t}_c]_\times \, {}^p\widehat{\mathbf{R}}_c \right) \widehat{\mathbf{L}}_\omega^{-1}$ and $\Delta^p\mathbf{t}_c = {}^p\mathbf{t}_c - {}^p\hat{\mathbf{t}}_c$

As in the planar case, described in Section 3.2.1, here it is clear that camera position ${}^p\mathbf{t}_c$ in the moving-platform frame $\mathcal{F}_p$ has no effect on the system stability. Indeed, it is only the difference between the actual camera position ${}^p\mathbf{t}_c$ and its estimation ${}^p\hat{\mathbf{t}}_c$ that has an effect of system stability.

Due to the high variability of the previous numerical analysis (Fig. 3.3), adding another two sets of spatial rotation makes such a graph too complex to be useful. Indeed, there are three rotations in this stability criterion: target object orientation $\theta\mathbf{u}$ in the camera frame; camera orientation ${}^p\mathbf{R}_c$ in the moving-platform frame; and the estimation ${}^p\widehat{\mathbf{R}}_c$ of the latter. As it was found in the stability analysis of the planar CDPR, all perturbations affect the stability criterion jointly. For this reason, there are indeed countless possible combinations of the three rotations within system stability.

Main observations as a result of the numerical analysis are:

i. The system will be stable as long as $\Delta \boldsymbol{e} \leqslant \boldsymbol{e}$.
ii. The lower $\Delta \boldsymbol{e}$, the larger the available rotational range.
iii. Finally, $\Delta^{p}\mathbf{t}_{c}$ does not affect system stability, as long as it is reasonably small (tested range was up to 0.2 m).

Note that the Jacobian matrix has not been considered as it is assumed to be precisely estimated. Accordingly, the stability criterion is affected only by the perturbations in hand-eye calibration, which is expressed as the transformation matrix ${}^{p}\mathbf{T}_{c}$, and by the visual servoing. It appears that the system stability is little sensitive to perturbations in hand-eye calibration. Therefore, the object pose estimation should be the primary focus to improve system behavior.

Stability Analysis III

Finally, the perturbations in all the matrices $\widehat{\mathbf{L}}_{s}$, $\widehat{\mathbf{A}}$, and $\widehat{\mathbf{A}}_{d}$ are taken into account. The stability criterion is kept in its full unsimplified form (3.1).

The effect of errors in the CDPR model on the stability of the system is investigated. The model of ACROBOT, the CDPR prototype shown in Fig. 1.9a is used. This model considers the Cartesian coordinates of exit points A_{i} and anchor points B_{i}, defined in Table A.1. It is assumed that $\Delta B_{i} = 0.008\,\text{m}$, meaning that the Euclidean distance between the actual anchor point B_{i} and its estimation $\widehat{B}_{i}$ is 0.008 m.

Some variable range examples to ensure system stability are shown in Fig. 3.4. Since the rotation range about z axis for a CDPR is significantly greater than about x or y axis, in this analysis the orientation of the moving-platform and its estimation has been represented with Euler angles. The following notation is used: θ, ϕ and ψ are Euler angles about x, y and z axes respectively.

Some noteworthy characteristics were observed in addition to (i)-(ii) of *Stability Analysis II* :

i. The smaller $\Delta^{b}\mathbf{t}_{p}$ or $\Delta^{p}\mathbf{t}_{c}$, the higher the range of rotation of the moving-platform, especially about z axis.
ii. The lower the rotation of the moving-platform about z axis, the higher its rotation about x and y axes.
iii. System stability is heavily dependent on moving-platform position in the base frame $\mathcal{F}_{b}$. The closer the moving-platform to the origin of $\mathcal{F}_{b}$, the larger $\Delta^{b}\mathbf{t}_{p}$ while keeping the system stable.
iv. Similarly, the closer the moving-platform to the origin of $\mathcal{F}_{b}$, the larger the range of rotation.

Perturbations affect the stability criterion jointly. By improving the knowledge of one parameter and thus reducing the expected level of error on this parameter, it is possible to increase the perturbation range of another parameter within system stability. As for a planar CDPR, here a correlation can be seen between the desired moving-platform

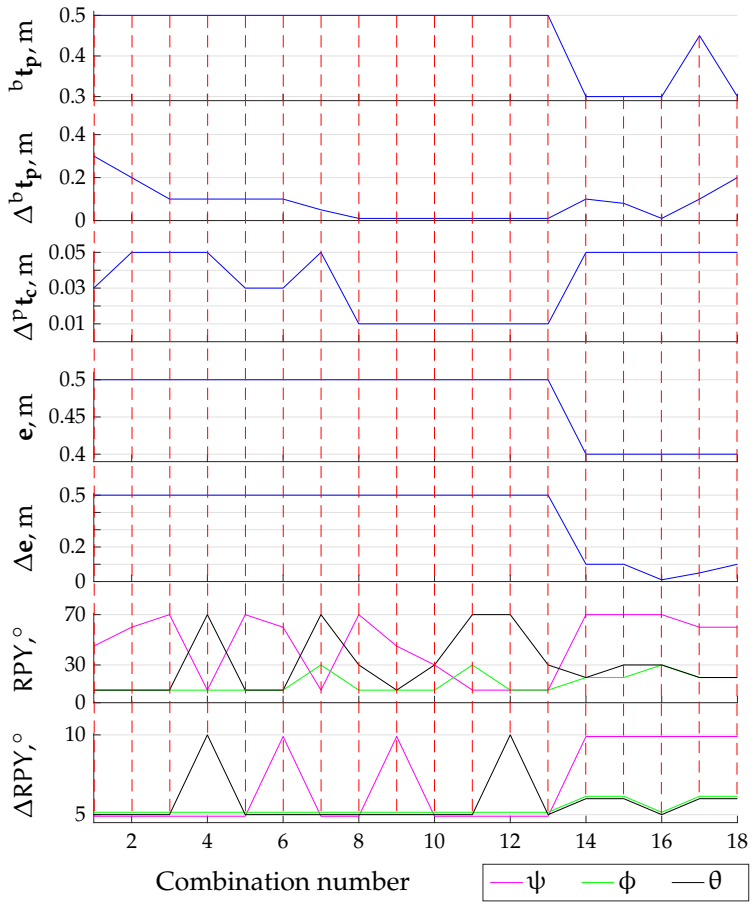


Figure 3.4: Example of possible combinations of design variables (${}^{b}\mathbf{t}_{p}$, $\Delta^{b}\mathbf{t}_{p}$, $\Delta^{p}\mathbf{t}_{c}$, $\mathbf{e}$, $\Delta\mathbf{e}$, ϕ, θ, ψ, $\Delta\phi$, $\Delta\theta$, $\Delta\psi$)

motion range and the perturbation size. For this reason, a novel workspace named Control Stability Workspace is defined and computed in Section 3.3.

The presented system is robust to large perturbations, including those in the CDPR model. The robustness to CDPR model errors is possibly one of the main advantages of visual servoing w.r.t model-based control.

3.2.3 Stability Analysis of a Spatial CDPR with IBVS

The control strategy for IBVS is the same as for PBVS, shown in Fig. 1.22. Of course the feature vector $\mathbf{s}$ no longer contains 3D features, but instead four points in the image are used as described in Section 1.4.3.

As mentioned before, when the feature vector consists of four points system stability is affected. In this case the stability criterion $\boldsymbol{\Pi}$, defined in (3.1), becomes a (8×8)-matrix that is at most of rank 6. Thus, the condition $\boldsymbol{\Pi} > \mathbf{0}$ cannot be held.

Only local asymptotic stability can be obtained for IBVS. To study it, we define a new error $\boldsymbol{e}'$ so that $\boldsymbol{e}' = \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\boldsymbol{e}$. The time derivative of $\boldsymbol{e}'$ is:

$$\dot{\boldsymbol{e}}' = \dot{\widehat{\mathbf{L}}}_{\mathbf{s}}^{\dagger}\,\boldsymbol{e} + \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\dot{\boldsymbol{e}} = \big(\mathbf{O} + \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\mathbf{L}_{\mathbf{s}}\big)^{\mathrm{c}}\mathbf{v}_{\mathrm{c}} \tag{3.21}$$

where $\mathbf{O}$ is equal to $\mathbf{0}$ when $\boldsymbol{e} = \mathbf{0}$ for any choice of $\mathbf{s}$ [CH08].

The model equation for IBVS of CDPRs then becomes:

$$\dot{\boldsymbol{e}}' = \big(\mathbf{O} + \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\mathbf{L}_{\mathbf{s}}\big)\boldsymbol{A}_{\mathrm{d}}^{-1}\boldsymbol{A}^{\dagger}\,\dot{\mathbf{l}} \tag{3.22}$$

Note that camera velocity ${}^{\mathrm{c}}\mathbf{v}_{\mathrm{c}}$ expressed as a function of the new error $\boldsymbol{e}'$ becomes:

$${}^{\mathrm{c}}\mathbf{v}_{\mathrm{c}} = -\lambda\widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\boldsymbol{e} = -\lambda\boldsymbol{e}' \tag{3.23}$$

Thus, the cable velocity vector can be expressed from (1.18), (1.37) and (3.23) as:

$$\dot{\mathbf{l}} = -\lambda\,\widehat{\mathbf{A}}\,\widehat{\mathbf{A}}_{\mathrm{d}}\,\boldsymbol{e}' \tag{3.24}$$

Finally, the closed-loop equation is expressed by injecting (3.24) into (3.22):

$$\dot{\boldsymbol{e}}' = -\lambda\big(\mathbf{O} + \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\mathbf{L}_{\mathbf{s}}\big)\boldsymbol{A}_{\mathrm{d}}^{-1}\boldsymbol{A}^{\dagger}\widehat{\mathbf{A}}\,\widehat{\mathbf{A}}_{\mathrm{d}}\,\boldsymbol{e}' \tag{3.25}$$

which is locally asymptotically stable in the neighborhood of $\mathbf{s} = \mathbf{s}^{*}$ if

$$\boldsymbol{\Pi}_{LAS} = \widehat{\mathbf{L}}_{\mathbf{s}}^{\dagger}\,\mathbf{L}_{\mathbf{s}}\,\boldsymbol{A}_{\mathrm{d}}^{-1}\boldsymbol{A}^{\dagger}\widehat{\mathbf{A}}\,\widehat{\mathbf{A}}_{\mathrm{d}} > \mathbf{0} \tag{3.26}$$

3.3 Control Stability Workspace

Before using a CDPR, one needs to know its workspace. Among the existing workspaces [SK06] [Ver04], the static feasible workspace (SFW of $\mathcal{S}$) that was introduced in Section 1.3.3 is a simple one. It is a kineto-static workspace that shows all the poses that the moving-platform is physically able to attain. In addition, it is important to evaluate the ability of the CDPR to reach a pose from a control perspective.

While analyzing the planar and spatial CDPRs in Section 3.2, it was concluded that the results of stability analysis are dependent on the moving-platform pose, because it shows up in the stability criterion $\boldsymbol{\Pi}$ through the Jacobian matrix $\mathbf{A}$, which is a function of the moving-platform pose. We showed that the closer the moving-platform to the origin of the base frame $\mathcal{F}_{\mathrm{b}}$, the larger the tolerated perturbations within system stability.

Accordingly, a novel workspace, named Control Stability Workspace (CSW), can be defined as follows:

$$\mathcal{Z} = \{\mathbf{p}_{\mathrm{p}} \in \mathrm{SE}(3) : \forall \mathbf{d} \in \mathcal{D}, \boldsymbol{\Pi} > \mathbf{0}\} \tag{3.27}$$

The workspace $\mathcal{Z}$ is the set of all moving-platform poses $\mathbf{p}_p$, for which the stability criterion $\mathbf{\Pi}$ is positive definite for any vector of perturbations $\mathbf{d}$ that is within bounds $\mathcal{D}$. It means that for any moving-platform pose within its CSW, the robot controller will be able to guide the moving-platform to its goal.

Note that the range combinations shown in Fig. 3.2 are examples of different definitions of perturbation bounds $\mathcal{D}$.

It is of interest to create a compound workspace, that takes into account the controller and the kineto-static performance of the robot. Indeed, on the one hand, a moving-platform pose can belong to $\mathcal{S}$ while being outside of $\mathcal{Z}$, namely, it is in a static equilibrium, but it may fail to reach the goal. On the other hand, a moving-platform pose can belong to $\mathcal{Z}$ while being outside of $\mathcal{S}$, namely, the robot controller will make the moving-platform reach the goal although the moving-platform is not in a static equilibrium (and thus it could be unable to remain in that pose). Thus we define a compound workspace, named $\mathcal{SZ}$, as the intersection of $\mathcal{S}$ and $\mathcal{Z}$:

$$\mathcal{SZ} = \{\mathbf{p}_p \in SE(3) : \exists \boldsymbol{\tau} \in \mathcal{T},\ \forall \mathbf{d} \in \mathcal{D},\ \boldsymbol{W}\boldsymbol{\tau} + \boldsymbol{w}_g = \mathbf{0}_6,\ \mathbf{\Pi} > \mathbf{0}\} \tag{3.28}$$

The compound workspace $\mathcal{SZ}$ is the set of all moving-platform poses $\mathbf{p}_p$ for which there exists a vector of cable tensions $\boldsymbol{\tau}$ within the cable tension space $\mathcal{T}$ such that the CDPR can balance the gravity wrench $\boldsymbol{w}_g$ leading to $\boldsymbol{W}\boldsymbol{\tau} + \boldsymbol{w}_g = \mathbf{0}_6$, and for which for any vector of perturbations $\mathbf{d}$ that is within bounds $\mathcal{D}$, the stability criterion $\mathbf{\Pi}$ is positive definite.

3.3.1 The Definition of Perturbation Bounds $\mathcal{D}$

As was shown in the stability analysis, there are many parameters that can be perturbed. However, it is possible to divide them in two categories: rotational and translational perturbations. Here, the physical meaning of these bounds is defined for a planar and a spatial CDPR.

Planar Case

Here, for the planar CDPR, we assume that the perturbations are also planar to be able to take them into account when computing the stability criterion $\mathbf{\Pi}$. Given a translational parameter $\mathcal{X}$, the translational perturbation bound is defined as a radius r_x of a circle, as shown in Fig. 3.5.

It signifies that the perturbation $\mathbf{d}_x$ of parameter $\mathcal{X}$ that is within the range $\mathcal{D}_x$ can have a magnitude of up to r_x and can be in any direction. Thus, the perturbed value of parameter $\mathcal{X}$ can be anywhere within the circle, e.g. it could take the value $\mathcal{X}_{d1}$ or $\mathcal{X}_{d2}$.

For any angle θ the perturbation bound is defined as $\Delta\theta$, as shown in Fig. 3.6. Thus, the perturbed value of the angle will be within the range: $\theta - \Delta\theta \leqslant \theta_d \leqslant \theta + \Delta\theta$.

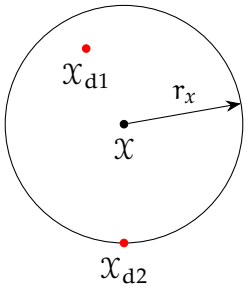


Figure 3.5: Translational parameter $\mathcal{X}$ and its perturbation radius r_x

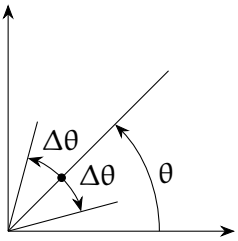


Figure 3.6: Parameter θ and its perturbation angle $\Delta\theta$

Spatial Case

In spatial case the perturbations are also spatial. For a position $\mathcal{X}$ the translational perturbation bound is defined as a radius r_x of a sphere, as shown in Fig. 3.7. Similarly to the planar case, any perturbation vector with origin at $\mathcal{X}$ and magnitude less than or equal to r_x belongs to this perturbation bound $\mathcal{D}_x$. Consequently, the perturbed value of $\mathcal{X}$ will be within the sphere defined by its origin at $\mathcal{X}$ and radius r_x. Two examples, $\mathcal{X}_{d1}$ and $\mathcal{X}_{d2}$, are shown in Fig. 3.7.

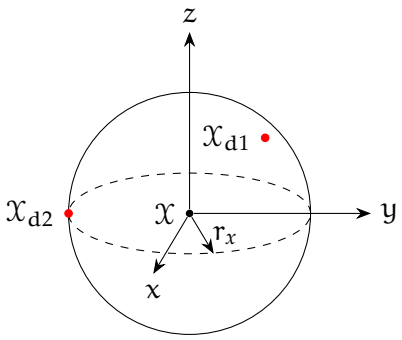


Figure 3.7: Translational parameter $\mathcal{X}$ and its perturbation radius r_x

For the rotational perturbation the axis-angle representation is used. The rotational perturbation bound is defined as the angle $\Delta\theta$ for all axes $\mathbf{u}$. That is a perturbation bound $\mathcal{D}_\theta$ contains all rotational perturbations with the angle less than or equal to $\Delta\theta$ and any axis $\mathbf{u}$.

3.3.2 CSW for a Planar CDPR with PBVS

As a continuation of the case study shown in Section 3.2.1, the Control Stability Workspace was computed for the planar CDPR and for different perturbation bounds $\mathcal{D}$. The planar CDPR is presented in Section 1.3.2.

First, the baseline perturbation bounds $\mathcal{D}_b$ is defined with the following values:

- moving-platform position error range $r_{bp} = 0.03\,\text{m}$
- moving-platform orientation error range $\Delta\theta_{bp} = 3^\circ$
- camera position error range $r_{pc} = 0.01\,\text{m}$
- camera orientation error range $\Delta\theta_{pc} = 3^\circ$
- cable exit point error range $r_{Ai} = 0.005\,\text{m}$
- cable anchor point error range $r_{Bi} = 0.005\,\text{m}$
- target object position error (feature vector error) $r_s = 0.01\,\text{m}$

These values are chosen based on the geometry of ACROBOT and taking into account some manufacturing errors that could be present on the robot. For example, there are pulley sheaves of 9 mm diameter on cable exit points that are not taken into account in the kinematic modeling of the CDPR. Similarly, cable anchor points are modeled as points, while they are actually located on a sphere of 0.008 m diameter about the modeled points. Camera position and orientation errors can also be amounted to manufacturing errors. Finally, the moving-platform position and orientation errors illustrate the possible coarseness of estimation of the moving-platform pose. Indeed, for many CDPRs the initial or *homing* moving-platform pose is often measured by hand and thus having a small error on the pose estimation is clearly possible. Furthermore, as was shown in Chapter 2 even when the initial moving-platform pose is measured before the experiment, the estimation of the moving-platform pose accumulates errors over time.

Note that the feature vector error consists of only the translational part, because the orientation of the target object is not present in the stability criterion $\boldsymbol{\Pi}_{pl}$ and thus a bad estimation of the orientation cannot make the condition $\boldsymbol{\Pi}_{pl} > \mathbf{0}$ to be not true.

During CSW computation, the workspace was discretized to a matrix of 33×33 points with X and Y coordinates ranging from $-0.48\,\text{m}$ to $0.48\,\text{m}$ and the step being $0.03\,\text{m}$. The area of a complete workspace is $0.92\,\text{m}^2$. Thus, 1089 moving-platform positions were tested for stability. For each of the moving-platform poses, the perturbation bound was applied as described in Section 3.3.1.

CSW was computed for $\mathcal{D}_b$ and is used as a reference. It is shown in every sub-figure of Fig. 3.8 in blue and its area is $0.878\,\text{m}^2$ or 95.4% of the full workspace. Then the range of each perturbation was increased, while keeping the rest as defined in $\mathcal{D}_b$. Every time CSW was created and its area was computed. The change of CSW area as a function of different perturbations is shown in Fig. 3.9, while the full results can be found in Table A.8 of Appendix A.5. Note that in Fig. 3.9 the CSW area is shown as a percentage of the maximum area of $0.92\,\text{m}^2$. Furthermore, some CSW examples are also shown in Fig. 3.8.

It can be observed that some perturbations have a higher impact on CSW area, while others affect it only marginally. Indeed, increasing camera position error r_{pc} expressed in $\mathcal{F}_p$ from 0.01 m to 0.20 m, only slightly decreases the workspace area from 0.878 m^2 to 0.849 m^2, as shown in Figs. 3.9a and 3.8d. Indeed, only 3.1% of workspace is lost by

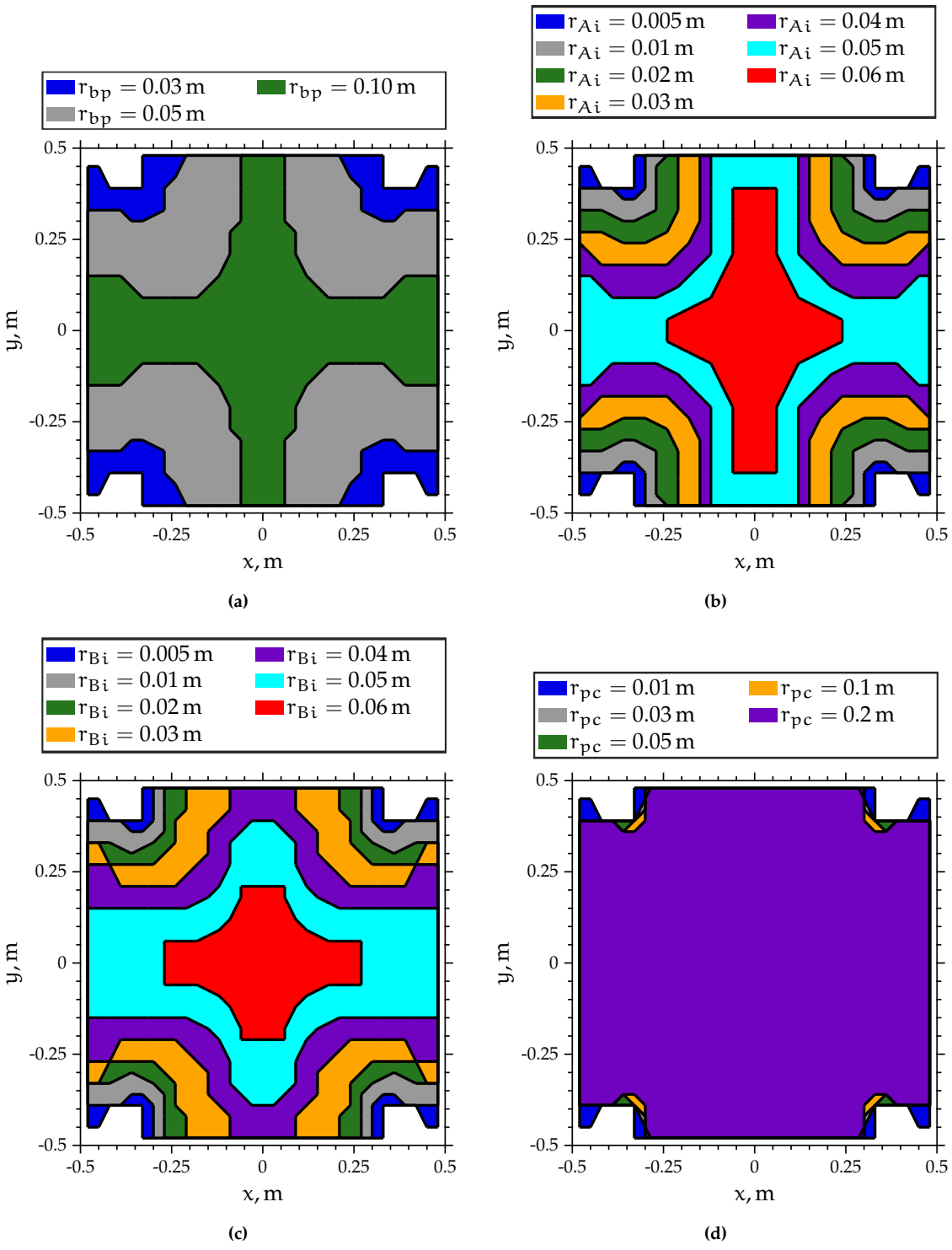


Figure 3.8: CSW for planar CDPR as a function of perturbation range

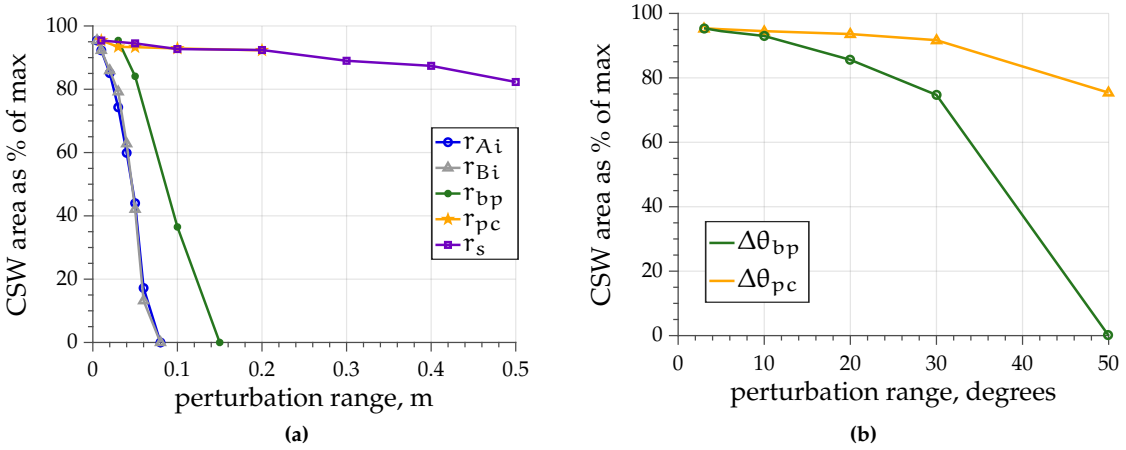


Figure 3.9: CSW area as a function of perturbation range

increasing r_{pc} twenty times. Furthermore, considering that the moving-platform size is 0.1 m×0.1 m, then it means that the camera can be anywhere on the moving-platform, without knowing where it is. The results are very similar for the translational error in object pose r_s, as can be seen in Fig. 3.9a, the curves for r_s and r_{pc} overlap. On the contrary, coarse moving-platform position estimation leads to a rapid reduction of CSW, as can be seen in Fig. 3.8a. By increasing the perturbation range to $r_{bp} = 0.1$ m, the CSW area is reduced to 0.336 m^2, which is only 36.5% of the full workspace. Similarly, errors in robot model, that is in cable exit and anchor points, have a significant influence on the workspace size, as can be seen in Figs. 3.8b, 3.8c and 3.9a.

Regarding rotational errors, the results are similar. That is, increasing the moving-platform rotational error $\Delta\theta_{bp}$ has a worse effect on the workspace area than the rotational error on camera pose $\Delta\theta_{pc}$, as can be seen in Fig. 3.9b.

It appears that the location of the parameter in the stability criterion determines how much its perturbation can affect the stability. Let us observe the stability criterion:

$$\boldsymbol{\Pi}_{pl} = \mathbf{L}_{s,pl}\,\boldsymbol{A}_{d,pl}^{-1}\,\boldsymbol{A}_{pl}^{\dagger}\,\widehat{\boldsymbol{A}}_{pl}\,\widehat{\boldsymbol{A}}_{d,pl}\widehat{\mathbf{L}}_{s,pl}^{-1} = \begin{bmatrix} -1 & 0 & e_y \\ 0 & -1 & -e_x \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} {}^{c}\mathbf{R}_p & -{}^{c}\mathbf{R}_p\mathbf{E}^{\top}{}^{p}\mathbf{t}_c \\ 0\ \ 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} {}^{p}\mathbf{u}_1^{\top} & {}^{p}\mathbf{b}_1^{\top}\mathbf{E}^{\top}{}^{p}\mathbf{u}_1 \\ \vdots & \vdots \\ {}^{p}\mathbf{u}_m^{\top} & {}^{p}\mathbf{b}_m^{\top}\mathbf{E}^{\top}{}^{p}\mathbf{u}_m \end{bmatrix}^{\dagger} \begin{bmatrix} {}^{p}\widehat{\mathbf{u}}_1^{\top} & {}^{p}\widehat{\mathbf{b}}_1^{\top}\mathbf{E}^{\top}{}^{p}\widehat{\mathbf{u}}_1 \\ \vdots & \vdots \\ {}^{p}\widehat{\mathbf{u}}_m^{\top} & {}^{p}\widehat{\mathbf{b}}_m^{\top}\mathbf{E}^{\top}{}^{p}\widehat{\mathbf{u}}_m \end{bmatrix} \begin{bmatrix} {}^{p}\widehat{\mathbf{R}}_c & \mathbf{E}^{\top}{}^{p}\widehat{\mathbf{t}}_c \\ 0\ \ 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & -\hat{e}_y \\ 0 & -1 & \hat{e}_x \\ 0 & 0 & -1 \end{bmatrix} \tag{3.29}$$

The robot model in the form of cable anchor and exit points as well as moving-platform pose are all in the Jacobian matrix $\boldsymbol{A}_{pl}$ and its estimation $\widehat{\boldsymbol{A}}_{pl}$. These are the parameters whose increased perturbation can rapidly make the system unstable. It should be noted

that the Jacobian and its estimation are right in the middle of Π_{pl} and it is the only matrix and its estimation pair that is directly multiplied by one another. The rest of the parameters are either in the adjoint matrix $\mathbf{A}_{d,pl}$ or in the interaction matrix $\mathbf{L}_{s,pl}$ and they appear to have very little effect on the stability criterion. Indeed, even when their perturbations are set to high values, such as $\Delta\theta_{pc} = 50°$ or $r_s = 0.5\,m$, the CSW remains large.

As an example, the CSW was also computed for two perturbation bounds found during the case study described in Section 3.2.1. More precisely, Cb. 2 and 16 are shown in Fig. 3.10 in gray and green, respectively. While workspace for Cb. 16 is almost full, the workspace for Cb. 2 is very limited. This is because almost all of the parameter perturbations have been increased in Cb. 2 when compared to Cb. 16. Note that leaving X or Y value close to 0 it would be possible to access the very border of CDPR workspace.

To conclude, visual servoing of a planar CDPR appears to be highly robust to many different perturbations and their combinations. However, significant errors in the robot model and bad moving-platform pose estimation can make the stability criterion Π_{pl} no longer be positive definite.

3.3.3 CSW for a Spatial CDPR with PBVS

CSW is computed for ACROBOT and CAROCA under PBVS control. Indeed, as the CDPR size is different, the workspace needs to be computed separately.

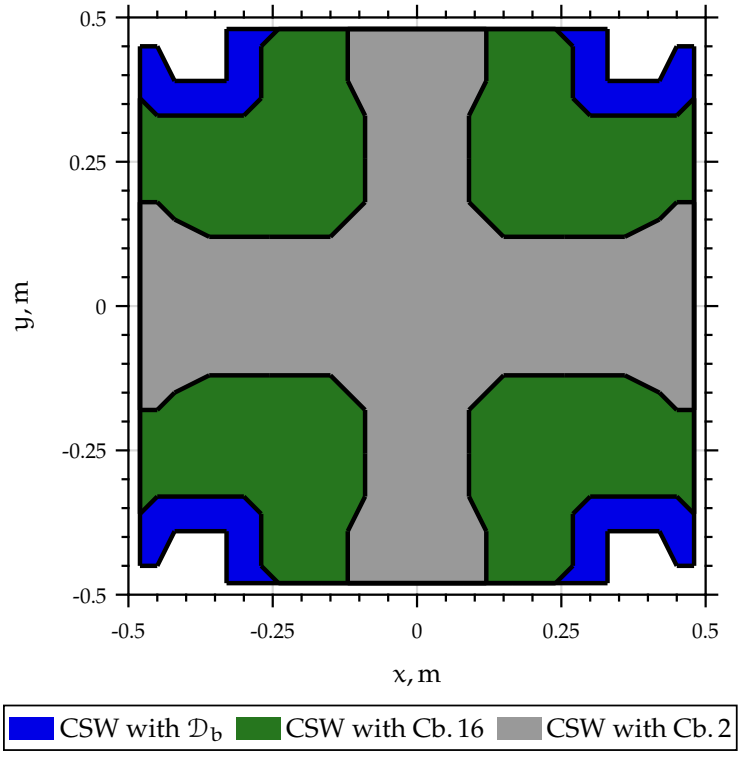


Figure 3.10: Control Stability Workspace example

The baseline perturbation bounds $\mathcal{D}_b$ are defined as follows:

- moving-platform position error range $r_{bp} = 0.03\,\text{m}$
- moving-platform orientation error range $\Delta\theta_{bp} = 3^\circ$
- camera position error range $r_{pc} = 0.01\,\text{m}$
- camera orientation error range $\Delta\theta_{pc} = 3^\circ$
- cable exit point error range $r_{Ai} = 0.005\,\text{m}$
- cable anchor point error range $r_{Bi} = 0.005\,\text{m}$

The bounds $\mathcal{D}_b$ are chosen to be the same for both robots for the sake of comparability. The baseline workspace is computed using $\mathcal{D}_b$ and then each of the perturbation ranges is increased incrementally.

CSW for ACROBOT with PBVS

For ACROBOT moving-platform positions are assigned as follows. The potential workspace is discretized to a set of 12 planes that are parallel to the global xy plane. Each plane is then discretized to a matrix of 33×33 points with X and Y coordinates ranging from $-0.48\,\text{m}$ to $0.48\,\text{m}$ and the step being $0.03\,\text{m}$. The Z coordinate of the planes ranges from $0\,\text{m}$ to $1.1\,\text{m}$ with the step being $0.1\,\text{m}$. Thus, the full workspace volume is $1.014\,\text{m}^3$. The workspace is computed for both moving-platform geometries that are presented in Appendix A.1.1. The evolution of the workspace volume is shown in Figs. 3.11 and 3.12 for the small and the large moving-platform geometries, respectively. Note that the vertical axis denotes the ratio of the current volume over the maximum volume of $1.014\,\text{m}^3$. Furthermore, the full results are shown in Appendix A.5 in Tables A.9 and A.10.

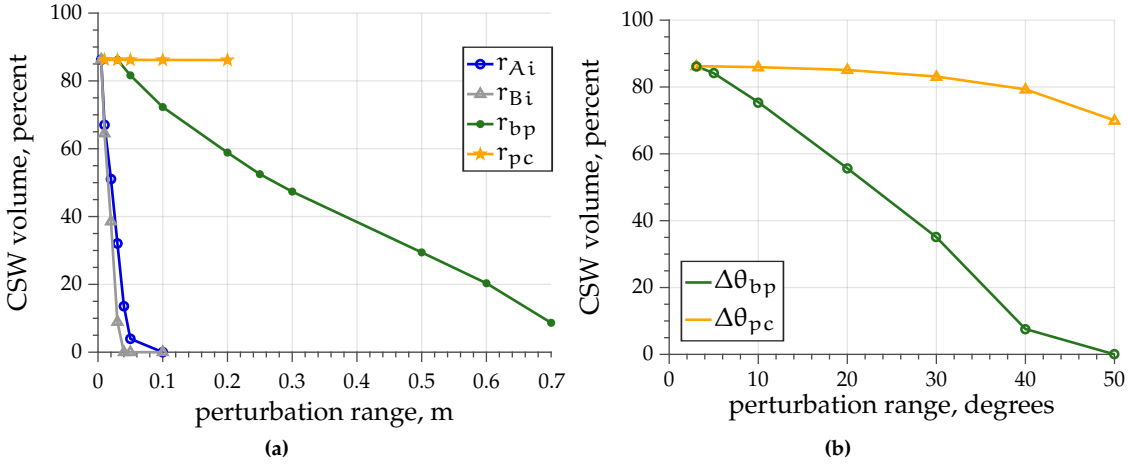


Figure 3.11: The effect of increasing perturbation range on CSW volume for ACROBOT with the small moving-platform and PBVS

First, the workspace corresponding to baseline perturbation bounds $\mathcal{D}_b$ is computed. For the small moving-platform the volume of CSW with $\mathcal{D}_b$ amounts to $0.875\,\text{m}^3$ or 86.3%. While for the large moving-platform the CSW volume, shown in Fig. 3.13a, is $0.88\,\text{m}^3$ or 87.0%. Clearly, at this stage the difference is not significant.

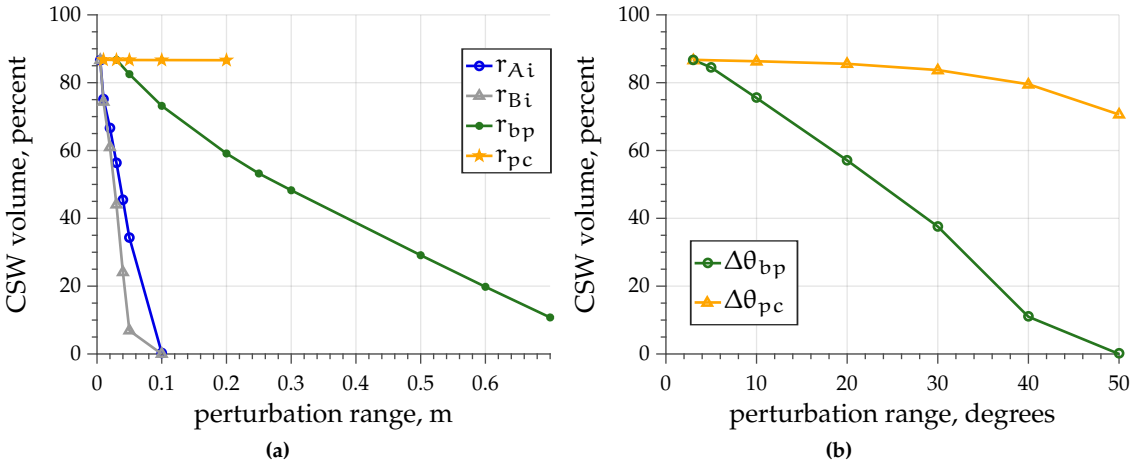
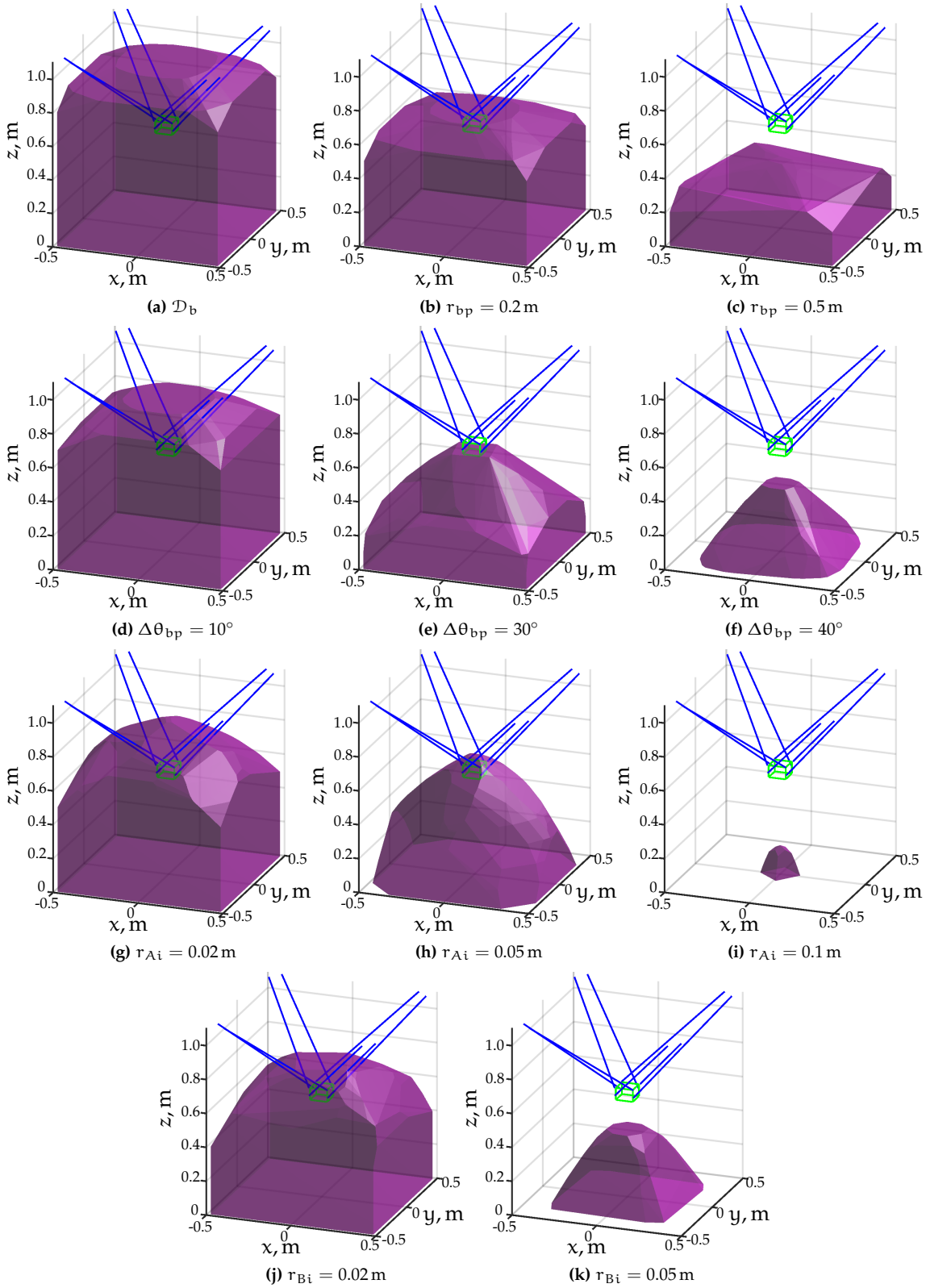


Figure 3.12: The effect of increasing perturbation range on CSW volume for ACROBOT with the large moving-platform and PBVS

Then the perturbation ranges are incrementally increased one by one, leaving the rest of perturbations as defined in $\mathcal{D}_b$. Some examples for the large moving-platform are shown in Fig. 3.13. It can be seen that all of the workspace reduction occurs in a downward direction along z axis no matter the perturbation. This is because ACROBOT is a suspended CDPR and its most stable position is ${}^b\mathbf{t}_p = [0\ \ 0\ \ 0]^\top$. Indeed, in Fig. 3.13i the tiny workspace is centered around ${}^b\mathbf{t}_p = [0\ \ 0\ \ 0]^\top$ with about 0.1 m displacement possible in any direction.

By comparing Figs. 3.11b and 3.12b, it is clear that the rotational perturbations have the same effect on the system, no matter the moving-platform geometry. Similarly, the increase r_{pc} and r_{bp} ranges leads to the same curves in Figs. 3.11a and 3.12a. However, this is not the case with errors on cable anchor and exit points. Indeed, for the smaller moving-platform, the CSW volume decreases significantly faster when r_{Bi} is increased. For example, for $r_{Bi} = 0.02\,\text{m}$, CSW volume for the large moving-platform is $0.618\,\text{m}^3$, while for the small moving-platform it is only $0.391\,\text{m}^3$. Then, for $r_{Bi} = 0.03\,\text{m}$ the workspace reduces to $0.447\,\text{m}^3$ and $0.091\,\text{m}^3$, respectively. Similarly, when r_{Ai} is increased, the CSW volume decreases more rapidly for the small moving-platform. Thus, it can be concluded that the geometry of the moving-platform influences the robustness of the system with respect to the CDPR modeling errors, namely, those in cable exit and anchor point coordinates.

As can be seen in Fig. 3.11a, the perturbations in cable exit and anchor points have the highest influence on system stability. Indeed, only 0.04 m error reduces the workspace by more than a half. However, if the error is no more than 0.01 m, then the workspace remains sufficiently large. For ACROBOT, the pulley diameter is 0.009 m as presented in Section 1.3.1. Indeed, the pulley radius is 0.0045 m and thus is lower than $r_{Ai} = 0.005\,\text{m}$ defined for $\mathcal{D}_b$. Thus, it is clear that for ACROBOT, the pulleys are small enough to be negligible.



(a) $\mathcal{D}_b$ **(b)** $r_{bp} = 0.2\,m$ **(c)** $r_{bp} = 0.5\,m$

(d) $\Delta\theta_{bp} = 10°$ **(e)** $\Delta\theta_{bp} = 30°$ **(f)** $\Delta\theta_{bp} = 40°$

(g) $r_{Ai} = 0.02\,m$ **(h)** $r_{Ai} = 0.05\,m$ **(i)** $r_{Ai} = 0.1\,m$

(j) $r_{Bi} = 0.02\,m$ **(k)** $r_{Bi} = 0.05\,m$

Figure 3.13: Visualization of CSW for ACROBOT with large moving-platform and PBVS

Finally, it appears that most of the CSW volume curves in Figs. 3.11 and 3.12 are almost straight lines, at least for the translational perturbations. Thus, it can be concluded that the

relationship between perturbation range and CSW volume is linear for the translational perturbations. The curve for rotational perturbation $\Delta\theta_{bp}$ is close to a straight line, while the one corresponding to $\Delta\theta_{pc}$ has a concave downward shape.

CSW for CAROCA with PBVS

CAROCA is a significantly larger CDPR than ACROBOT. The potential workspace is discretized to a set of 11 planes parallel to the global $\mathbf{xy}$ plane. Each of these planes is discretized to a matrix of 33×33 points. The X coordinate ranges from $-1.98\,\text{m}$ to $1.98\,\text{m}$ with the step being $0.12\,\text{m}$. The Y coordinate ranges from $-3.465\,\text{m}$ to $3.465\,\text{m}$, with the step being $0.21\,\text{m}$. Finally, the Z coordinate of planes ranges from $0\,\text{m}$ to $3\,\text{m}$ with the step being $0.3\,\text{m}$. This makes the full workspace volume equal to $82.33\,\text{m}^3$.

The CSW computed for CAROCA with baseline perturbation bounds $\mathcal{D}_b$ has a volume of $63.06\,\text{m}^3$, which is 76.6% of the full workspace. The change of the CSW volume depending on perturbation range is shown in Fig. 3.14, while some examples are shown in Fig. 3.15. Note that in Fig. 3.14 the percentage shown is the obtained volume for a given perturbation set over the maximum volume of $82.33\,\text{m}^3$. The full results can be found in Table A.12 in Appendix A.5. In general, the shape of the curves in Fig. 3.14 is similar to those in Figs. 3.11 and 3.12 for ACROBOT. More precisely, perturbation in CDPR model has the highest effect on CSW volume, following by moving-platform pose estimation and finishing with eye-hand calibration errors. However, in Fig. 3.15a the green curve of the translational error r_{bp} has an exponential shape. Furthermore, with the increased CDPR size, the range of this perturbation has increased. Even having set $r_{bp} = 1\,\text{m}$ does not make the workspace null, instead it is equal to approximately 10% of the full workspace and appears to be almost flat, as can be seen in Fig. 3.15d. Similarly, having $r_{Bi} = 0.1\,\text{m}$ or $r_{Ai} = 0.1\,\text{m}$ does not make the CSW non-existent for CAROCA. Instead, approximately 30% and 50% of the full workspace remains available, respectively, as can be seen in Figs 3.15j and 3.15l. Furthermore, even setting the camera position error range $r_{pc} = 0.5\,\text{m}$ has little effect on the CSW size, which remains at 71.9%. On

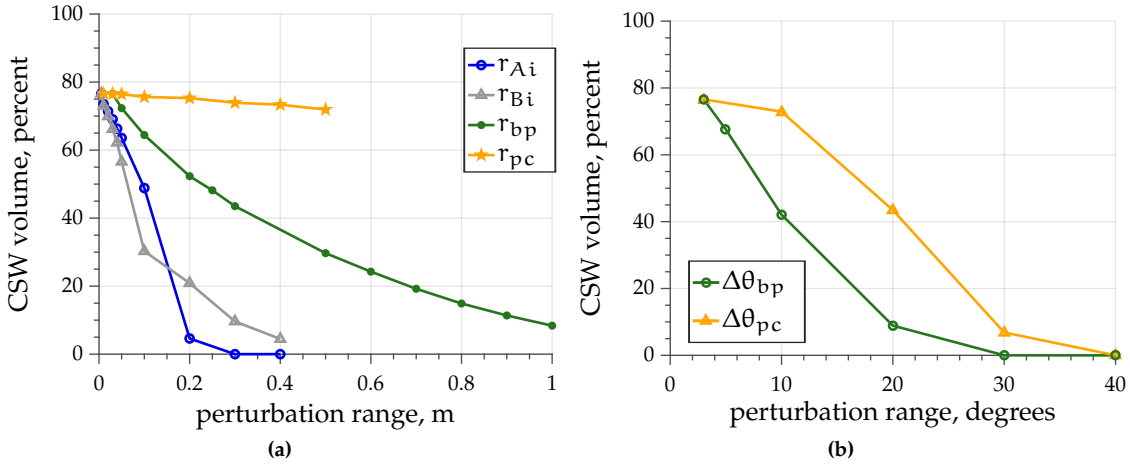
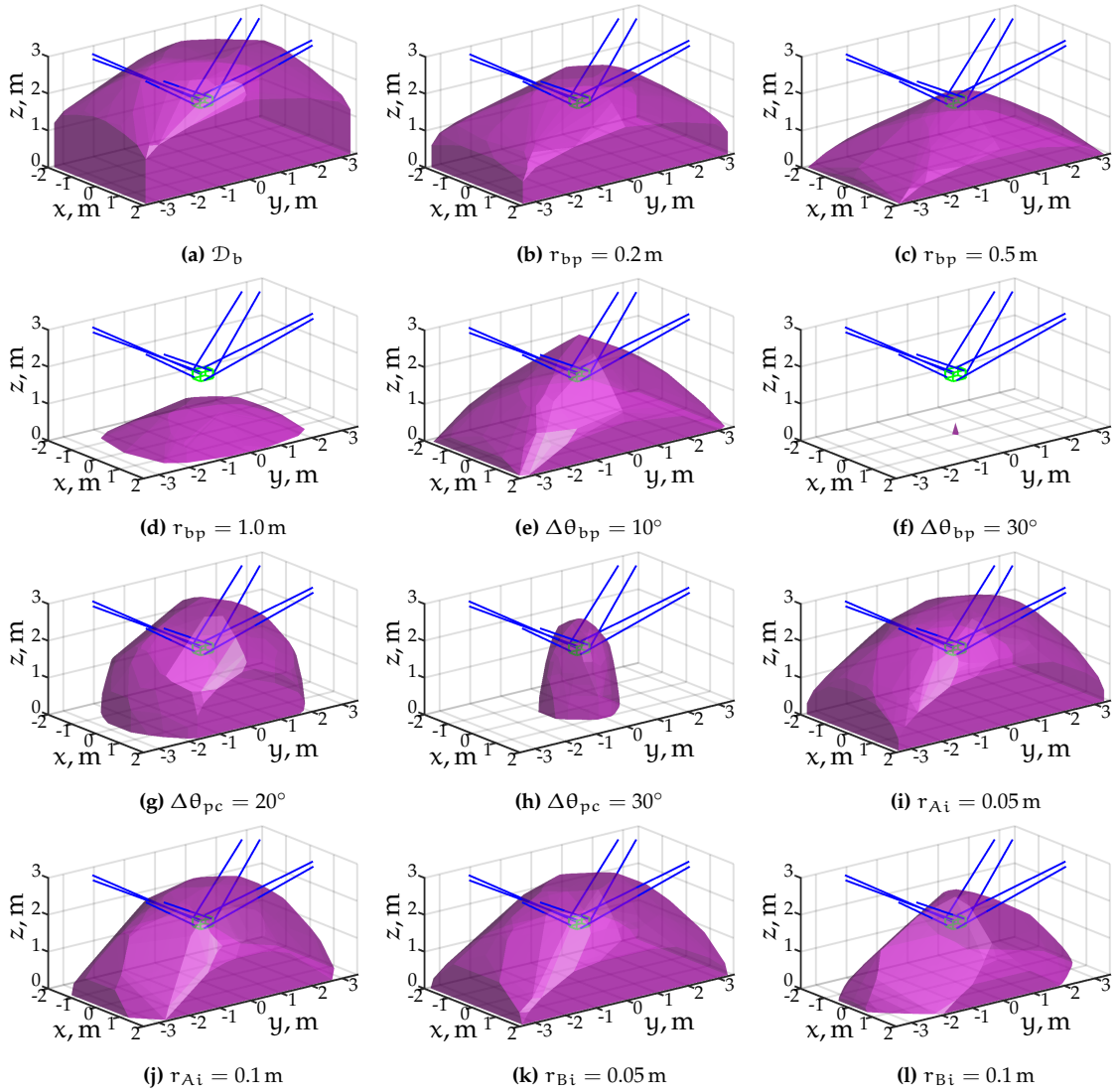


Figure 3.14: The effect of increasing perturbation range on CSW volume for CAROCA with PBVS

the other hand, rotational errors rapidly reduce the workspace. Especially those in the moving-platform pose estimation, where at $\Delta\theta_{bp} = 10°$ the volume is at 42.1% (Fig. 3.15e) and at $\Delta\theta_{bp} = 30°$ the volume is only at 0.003% (Fig. 3.15f). While the rotational error $\Delta\theta_{pc}$ on the camera pose also has a significant effect on the CSW size, however having $\Delta\theta_{pc} = 30°$ allows us to use 43.4% of the workspace (Fig. 3.15h), which is significantly larger than for $\Delta\theta_{bp}$ of the same size.

Having such a large CDPR also allows us to compare the change of CSW shape when some perturbations are increased. For example, the increase of r_{bp} is shown in Figs. 3.15a through 3.15d and appears to flatten the workspace by greatly reducing the range along z axis before affecting the workspace range in xy plane. On the other hand, rotational errors in eye-hand calibration appears to affect the xy range more severely. This can be seen by comparing Fig. 3.15a with 3.15g and 3.15h.



(a) $\mathcal{D}_b$ **(b)** $r_{bp} = 0.2\,m$ **(c)** $r_{bp} = 0.5\,m$

(d) $r_{bp} = 1.0\,m$ **(e)** $\Delta\theta_{bp} = 10°$ **(f)** $\Delta\theta_{bp} = 30°$

(g) $\Delta\theta_{pc} = 20°$ **(h)** $\Delta\theta_{pc} = 30°$ **(i)** $r_{Ai} = 0.05\,m$

(j) $r_{Ai} = 0.1\,m$ **(k)** $r_{Bi} = 0.05\,m$ **(l)** $r_{Bi} = 0.1\,m$

Figure 3.15: Visualization of CSW for CAROCA with PBVS

CSW for a Fully-Constrained CDPR

In a fully-constrained CDPR some of the cable pulleys are located on the ground. Due to this, cables are pulling on the moving-platform from all directions, which greatly increases its stiffness and velocity range. While both of the presented CDPRs are used in the suspended configuration, it is possible to reconfigure them in a fully-constrained configuration. In this section ACROBOT is used and four of its cables are reconfigured to exit from pulleys located on the ground. The modified cable exit point coordinates are shown in Table 3.1.

Table 3.1: ACROBOT cable exit point coordinates for the fully-constrained configuration

Cable exit points expressed in $\mathcal{F}_b$, m			
A1	$[0.52; 0.475; 0.0]^\top$	A5	$[-0.52; -0.465; 0.0]^\top$
A2	$[-0.47; 0.53; 1.165]^\top$	A6	$[0.47; -0.52; 1.165]^\top$
A3	$[-0.52; 0.475; 0.0]^\top$	A7	$[0.525; -0.47; 0.0]^\top$
A4	$[0.47; 0.53; 1.165]^\top$	A8	$[-0.47; -0.525; 1.165]^\top$

The evolution of the CSW size is shown in Fig. 3.16 and the full results can be found in Table A.11 in the Appendix A.5. Some examples of CSW visualization are shown in Fig. 3.17. As it can be seen in Fig. 3.17a, the workspace with $\mathcal{D}_b$ is quite large, however increasing the perturbation range leads to a rapid decrease in the workspace volume.

Similarly as with the suspended CDPR, the perturbation r_{pc} on the camera position with respect to the moving-platform frame appears to have no effect on the workspace size. Indeed, by increasing the perturbation from $r_{pc} = 0.01\,\text{m}$ to $r_{pc} = 0.2\,\text{m}$ the workspace volume changes from $0.985\,\text{m}^3$ to $0.983\,\text{m}^3$, as shown in Fig. 3.16a. The tolerance of rotational error $\Delta\theta_{pc}$ is significantly larger than for $\Delta\theta_{bp}$ (see Fig. 3.16b). Indeed, even with $\Delta\theta_{pc} = 30°$ the workspace is still almost at its maximum size and then starts to decrease. The CSW volume is still above 70% with $\Delta\theta_{pc} = 50°$. On the

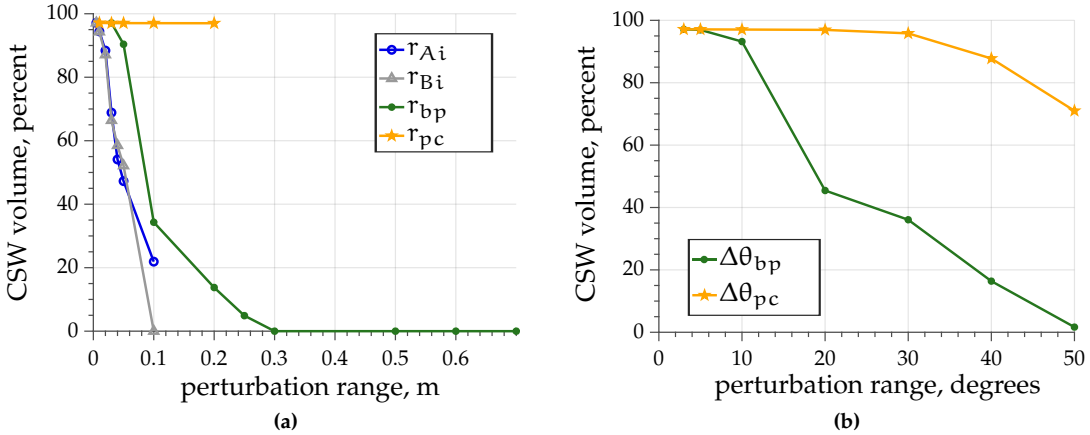


Figure 3.16: The effect of increasing perturbation range on CSW volume for fully-constrained ACROBOT with PBVS

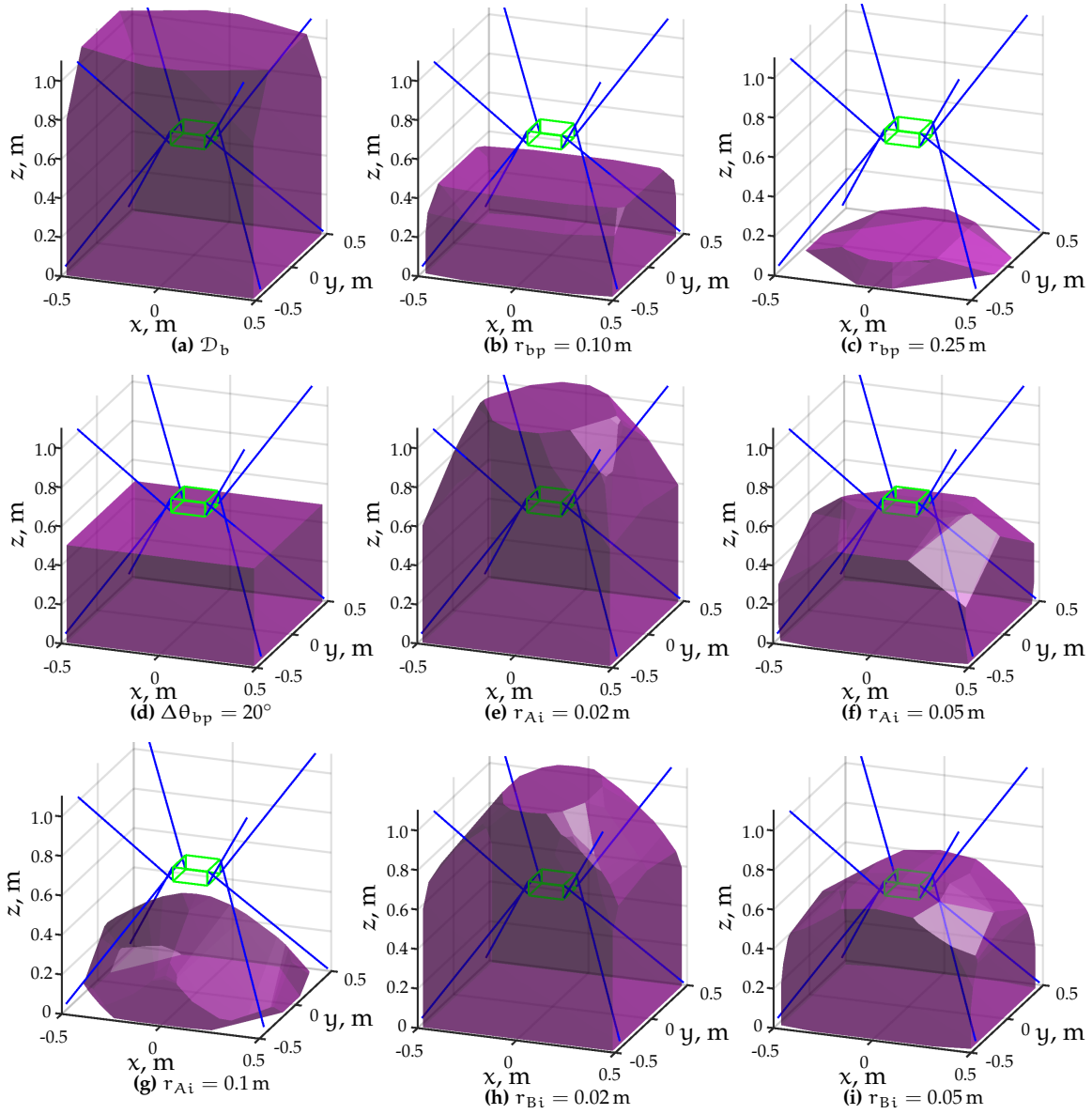


Figure 3.17: CSW volume visualization for fully-constrained ACROBOT with perturbations on multiple variables

other hand, increasing $\Delta\theta_{bp}$ above 10° leads to a rapid decrease of CSW volume. For example, with $\Delta\theta_{bp} = 20°$ the CSW volume, shown in Fig. 3.17d, is only at 45.4%. The workspace of a fully-constrained CDPR is still most affected by perturbations in cable exit point coordinates A_i and anchor point coordinates B_i, as shown in Fig. 3.16a. Unlike the suspended CDPR, here the errors in the moving-platform position reduce the workspace almost as rapidly as model errors. Indeed, as can be seen in Fig. 3.17b, with $r_{bp} = 0.10\,m$ the workspace is already quite small, its volume being $0.348\,m^3$. And then with $r_{bp} = 0.25\,m$ the workspace volume is only $0.05\,m^3$, which is only 5% of the full workspace size. Thus, it can be concluded that a CDPR in a fully-constrained configuration is significantly more sensitive to moving-platform pose errors.

The computation of CSW is rather strict, meaning that for a given moving-platform pose the perturbation needs to be validated in all the directions. Thus, even if in just one direction the perturbation leads to the stability criterion not being held, that moving-platform pose is marked as out of the CSW. Furthermore, it appears that the simple CDPR model with straight inelastic cables might be insufficient to evaluate the stability of a fully-constrained CDPR.

3.3.4 CSW for a Spatial CDPR with 2½D VS

Here, the CSW is computed for 2½D VS used on ACROBOT and CAROCA using the same initial perturbation bound $\mathcal{D}_b$ and perturbation augmentation as defined for PBVS in Section 3.3.3.

CSW for ACROBOT with 2½D VS

The description of CSW computation for ACROBOT is presented in the corresponding section for PBVS, namely Section 3.3.3. The results of CSW computation for ACROBOT with 2½D VS control are presented in this section. The workspace is computed for both moving-platform geometries, described in Appendix A.1, and the change of the workspace volume can be seen in Figs. 3.18 and 3.19, while the full results can be found in Tables A.13 and A.14 of Appendix A.5. In Figs. 3.18 and 3.19 the percentage shown is the ratio of the computed CSW volume for the given perturbation set over the maximum volume of 1.014 m^3 for ACROBOT. Furthermore, CSW volume visualization for the large moving-platform and different perturbation sets is shown in Fig. 3.20.

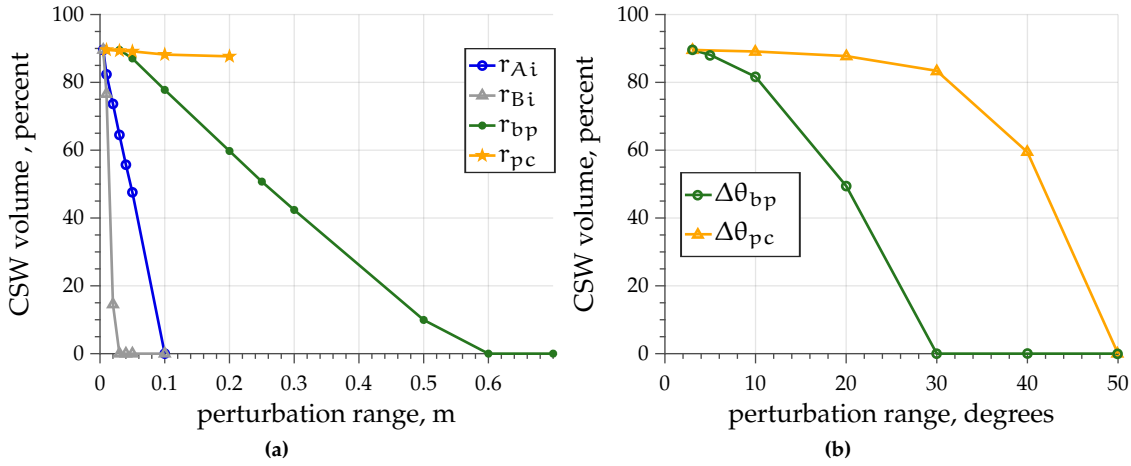


Figure 3.18: The effect of increasing perturbation range on CSW volume for ACROBOT with the small moving-platform and 2½D VS

First, the results for two moving-platform geometries are compared. Clearly, there is no significant difference with respect to rotational errors, as can be seen in Figs. 3.18b and 3.19b. Similarly, change of r_{pc} has the same negligible effect in both cases. Furthermore, the curve of r_{bp} appears to be the same in Figs. 3.18a and 3.19a. However, the CDPR model errors, namely r_{Ai} and r_{Bi} have a very different effect. Indeed, with the larger moving-platform 40.2% of the workspace remain available with $r_{Ai} = 0.1$ m,

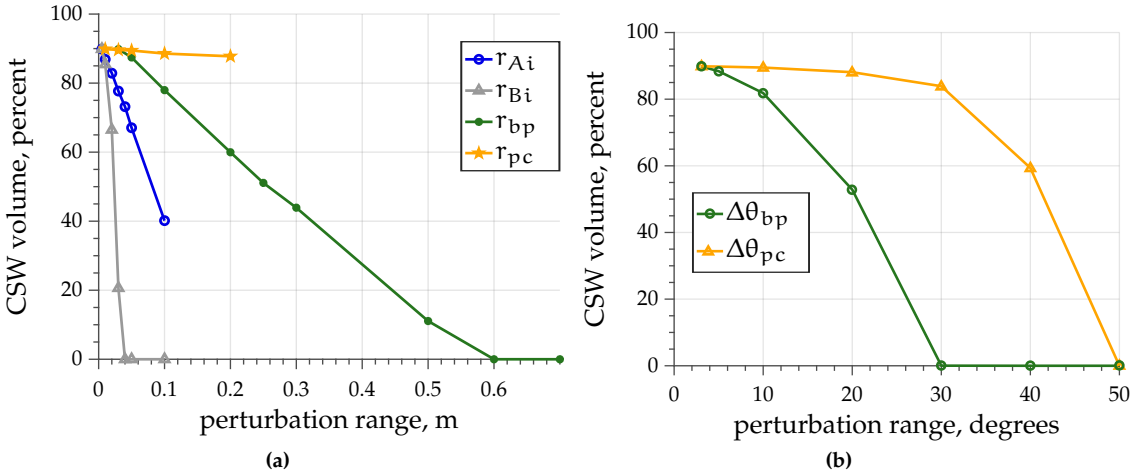


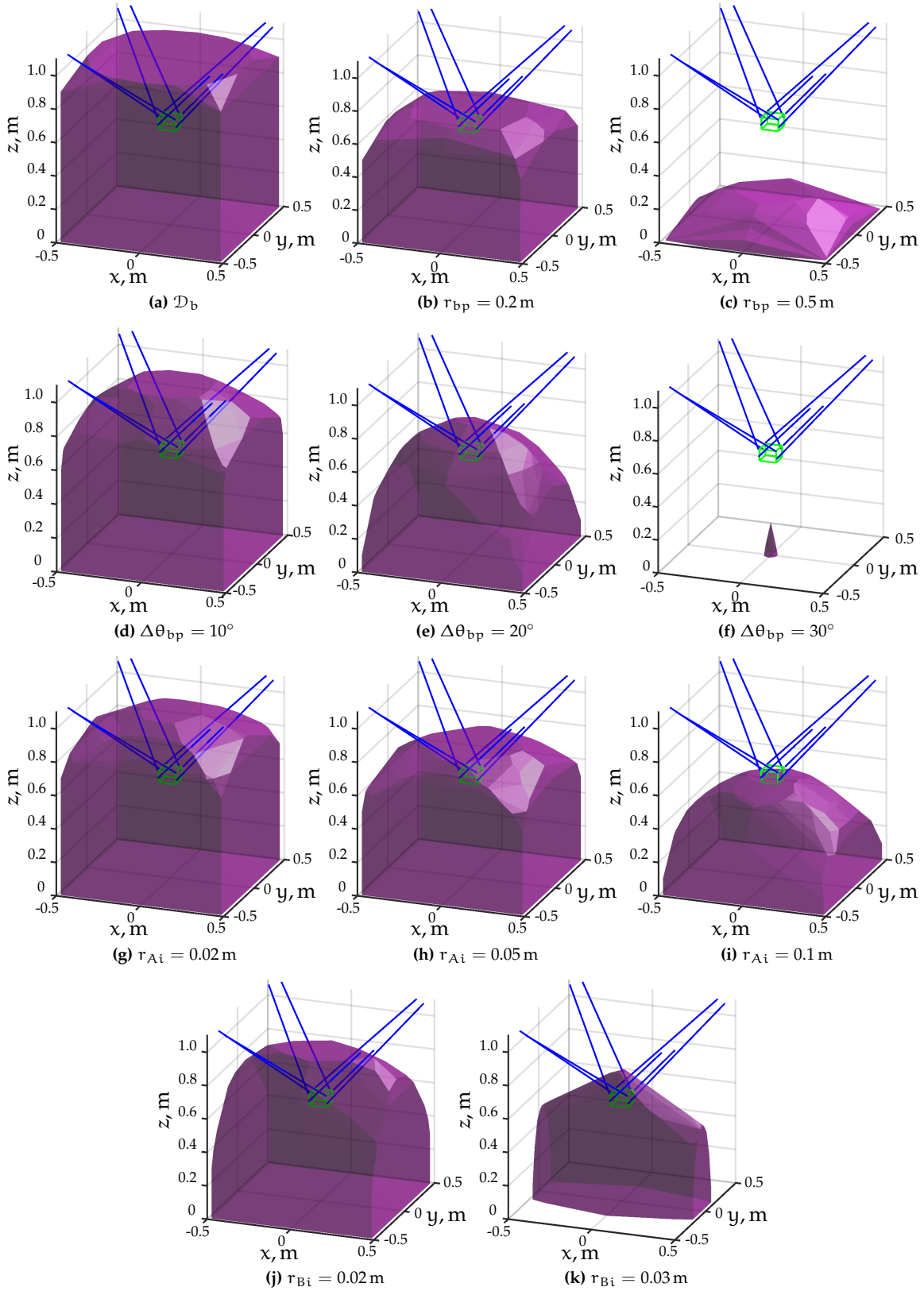
Figure 3.19: The effect of increasing perturbation range on CSW volume for ACROBOT with the large moving-platform and 2½D VS

while for the smaller moving-platform the CSW is non-existent for this value. Indeed, at $r_{Ai} = 0.05\,\text{m}$ the workspace is already at only 47.6% for the small moving-platform. Similarly, having $r_{Bi} = 0.02\,\text{m}$ leads to CSW being at 14.5% for the small moving-platform and at 66.5% for the large one. Thus, for 2½D VS same as for PBVS, increasing the moving-platform size allows us to increase the robustness to CDPR model errors.

It is also possible to compare the CSW volume for same moving-platform geometry and different VS controllers. When baseline perturbation set $\mathcal{D}_b$ is used, the resulting CSW is almost the same for PBVS and for 2½D VS, as can be seen in Figs. 3.13a and 3.20a. Indeed, for 2½D VS it is only 2.9% larger. Comparing the overall trend, it appears that 2½D VS is more robust to smaller perturbations, leading to a noticeably larger workspace when the perturbation range is rather low. But as the perturbation range is increased, the CSW volume decreases rapidly. For example, the CSW with $\Delta\theta_{bp} = 10°$ is significantly larger for 2½D VS, shown in Fig. 3.20d, than for PBVS, shown in Fig. 3.13d. However, setting $\Delta\theta_{bp} = 30°$ leads to a very small CSW for 2½D VS as shown in Fig. 3.20f, when compared to PBVS, shown in Fig. 3.13e. Similarly, the CSW for $r_{bp} = 0.5\,\text{m}$ for 2½D VS is smaller than for PBVS as can be seen in Figs. 3.20c and 3.13c, respectively, while it was larger before reaching this perturbation range.

2½D VS appears to be more robust to perturbation in the cable exit points. Indeed, at $r_{Ai} = 0.1\,\text{m}$ the CSW is at 40.2%, as can be seen in Fig. 3.20i, while it is only at 0.3% for PBVS, shown in Fig. 3.13i. However, note that the corresponding blue curve in Fig. 3.19a is steep; in fact it is only slightly less steep than the one in Fig. 3.18a for the small moving-platform. Thus, even though at $r_{Ai} = 0.1\,\text{m}$ the workspace is still large, it is clear that it decreases very rapidly.

On the other hand, the robustness to perturbation in cable anchor points diminishes even more rapidly. That is, for $r_{Bi} = 0.02\,\text{m}$ the CSW of 2½D VS is larger, but it becomes



(a) $\mathcal{D}_b$ **(b)** $r_{bp} = 0.2\,m$ **(c)** $r_{bp} = 0.5\,m$

(d) $\Delta\theta_{bp} = 10°$ **(e)** $\Delta\theta_{bp} = 20°$ **(f)** $\Delta\theta_{bp} = 30°$

(g) $r_{Ai} = 0.02\,m$ **(h)** $r_{Ai} = 0.05\,m$ **(i)** $r_{Ai} = 0.1\,m$

(j) $r_{Bi} = 0.02\,m$ **(k)** $r_{Bi} = 0.03\,m$

Figure 3.20: Visualization of CSW for ACROBOT with large moving-platform and 2½D VS

non-existent at $r_{Bi} = 0.04\,m$, while some small workspace still exists for PBVS even at $r_{Bi} = 0.05\,m$.

CSW for CAROCA with 2½D VS

The description of CSW computation for CAROCA is presented in the corresponding section for PBVS, namely Section 3.3.3. The results of CSW computation for CAROCA with 2½D VS control are presented in this section. The change of the workspace volume can be seen in Fig. 3.21. Here the ratio of the CSW volume for the given perturbation set over the full workspace size of $82.33\,m^3$ is given. The full results can be found in Table A.15 of Appendix A.5. Furthermore, CSW volume visualization for different perturbation sets is shown in Fig. 3.22.

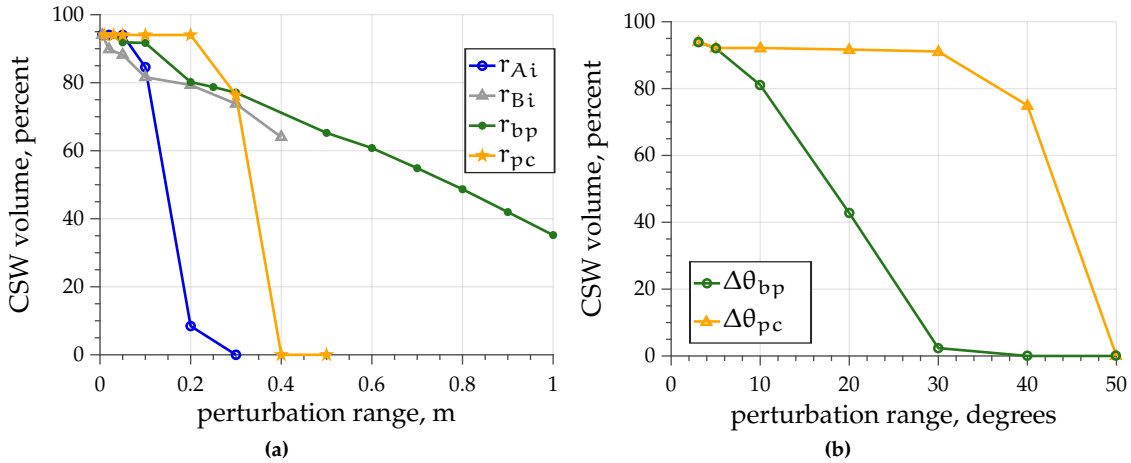
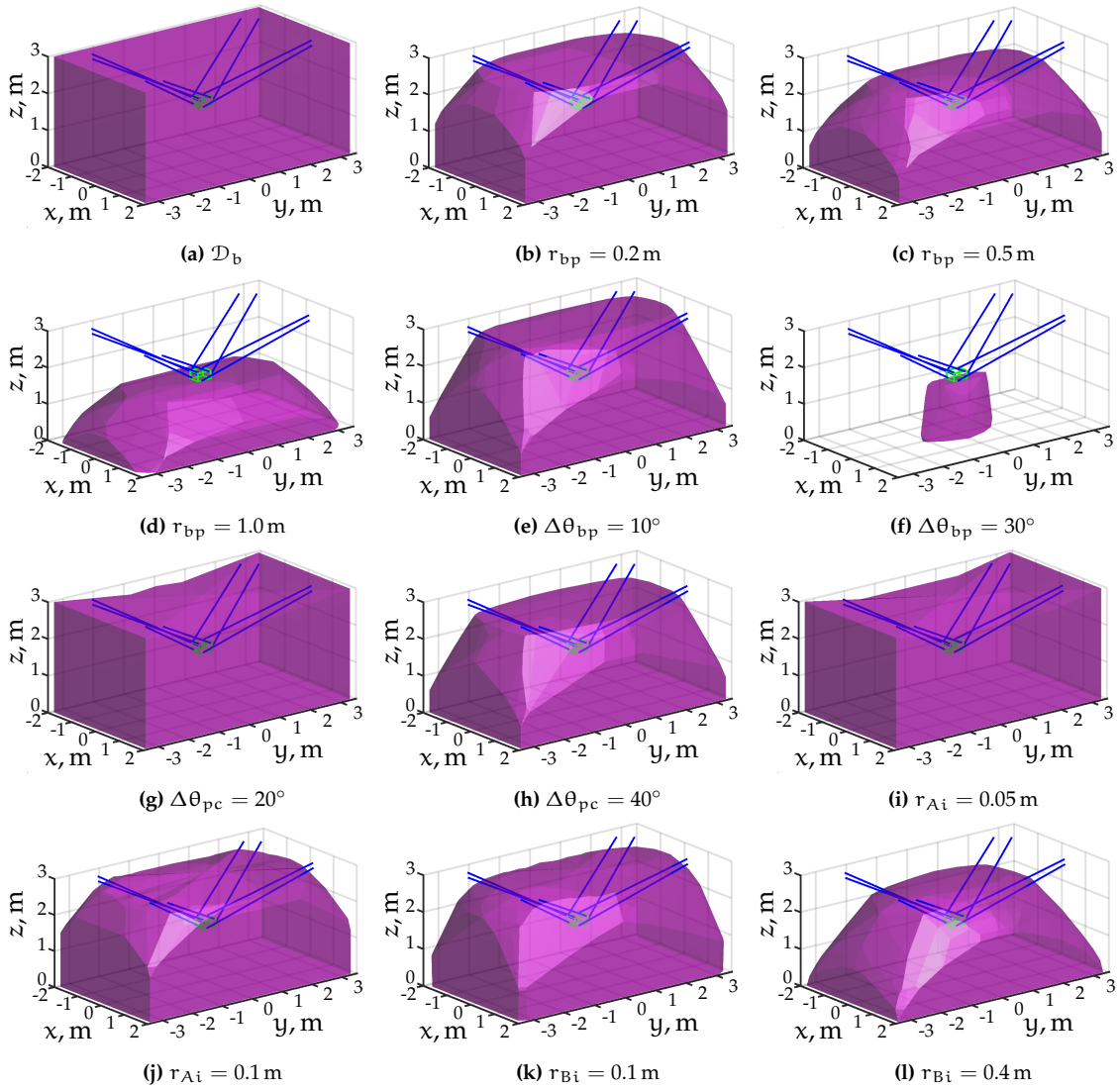


Figure 3.21: The effect of increasing perturbation range on CSW volume for CAROCA with 2½D VS

As for PBVS, the larger CDPR has a greater perturbation range. The moving-platform position error can go even up to 1 m with 30% of the full workspace remaining available. In comparison to PBVS, here a larger size of workspace is available for the same size of perturbation. This was true for ACROBOT, however it is even more pronounced for CAROCA. For example, for baseline perturbation bounds $\mathcal{D}_b$ 94% of the full workspace are available with 2½D VS, while only 76.6% are available with PBVS. On the other hand, the change of workspace size is more abrupt. With $r_{Ai} = 0.1\,m$ CSW is at 84.6%, while with $r_{Ai} = 0.2\,m$ CSW is suddenly only at 8.4%. Similarly, for the camera pose in the moving-platform frame the workspace goes from 94% to 76.4% to 0% by increasing $r_{pc} = 0.2\,m$ to $r_{pc} = 0.3\,m$ to $r_{pc} = 0.4\,m$. Though it should be noted that, given the moving-platform size, having $r_{pc} = 0.2\,m$ means that the camera position on the moving-platform does not need to be known. Also, note that this is the only combination of a visual servoing approach and a CDPR geometry where some value of r_{pc} has not just a visible effect on the CSW volume, but can actually make the workspace empty.



(a) $\mathcal{D}_b$ **(b)** $r_{bp} = 0.2\,m$ **(c)** $r_{bp} = 0.5\,m$

(d) $r_{bp} = 1.0\,m$ **(e)** $\Delta\theta_{bp} = 10°$ **(f)** $\Delta\theta_{bp} = 30°$

(g) $\Delta\theta_{pc} = 20°$ **(h)** $\Delta\theta_{pc} = 40°$ **(i)** $r_{Ai} = 0.05\,m$

(j) $r_{Ai} = 0.1\,m$ **(k)** $r_{Bi} = 0.1\,m$ **(l)** $r_{Bi} = 0.4\,m$

Figure 3.22: Visualization of CSW for CAROCA with 2½D VS

3.3.5 CSW for a Spatial CDPR with IBVS

As described in Section 3.2.3, only the local asymptotic stability can be evaluated for IBVS, because there are more features in the vector $\mathbf{s}$ than DoF of the moving-platofrm. Thus, the computed CSW is more an illustration and not a proof. Indeed, for most of the trajectory the system stability will be unknown.

CSW for ACROBOT with IBVS

The results of CSW computation for ACROBOT with IBVS control are presented in this section. The change of the workspace volume as the perturbation range is increased is shown in Figs. 3.23 and 3.24, while the full results can be found in Tables A.16 and A.17. In Figs. 3.23 and 3.24 it is the ratio between the CSW volume corresponding to a given perturbation set and the full workspace volume of $1.014\,m^3$ that is shown. Moreover, CSW volume visualization for the large moving-platform is shown in Fig. 3.25.

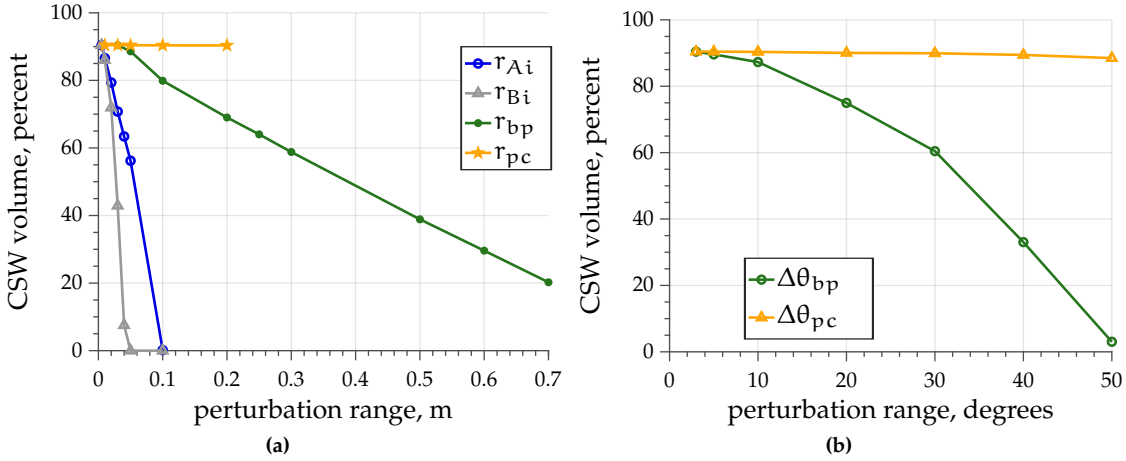


Figure 3.23: The effect of increasing perturbation range on CSW volume for ACROBOT with the small moving-platform and IBVS

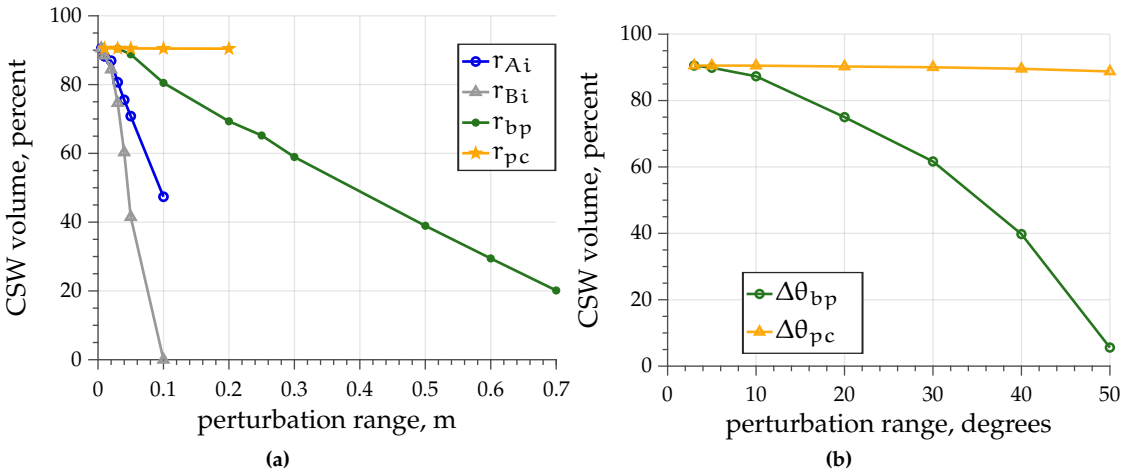


Figure 3.24: The effect of increasing perturbation range on CSW volume for ACROBOT with the large moving-platform and IBVS

Overall the curves in Figs. 3.23 and 3.24 show the same tendency as for PBVS and 2½D VS. However, the workspace volume reduction rate is noticeably slower. For example with $r_{bp} = 0.5\,\text{m}$ CSW for IBVS is $0.395\,\text{m}^3$ (Fig. 3.25c), while it is only $0.112\,\text{m}^3$ for 2½D VS (Fig. 3.20c) and $0.295\,\text{m}^3$ for PBVS (Fig. 3.13c). Similarly with $r_{Ai} = 0.1\,\text{m}$, the workspace measures $0.480\,\text{m}^3$, $0.407\,\text{m}^3$ and $0.003\,\text{m}^3$ for IBVS, 2½D VS and PBVS, respectively (see Figs. 3.25i, 3.20i and 3.13i, respectively). Note that the shape of the workspace changes in the same way for each perturbation no matter the visual servoing control. For example, increasing r_{bp} decreases the height of the CSW, but the corners remain within the workspace. On the contrary, increasing $\Delta\theta_{bp}$ the CSW shape becomes like a pyramid or a cone (especially in Fig. 3.20f). Increasing r_{Bi} the workspace flattens towards the yz plane.

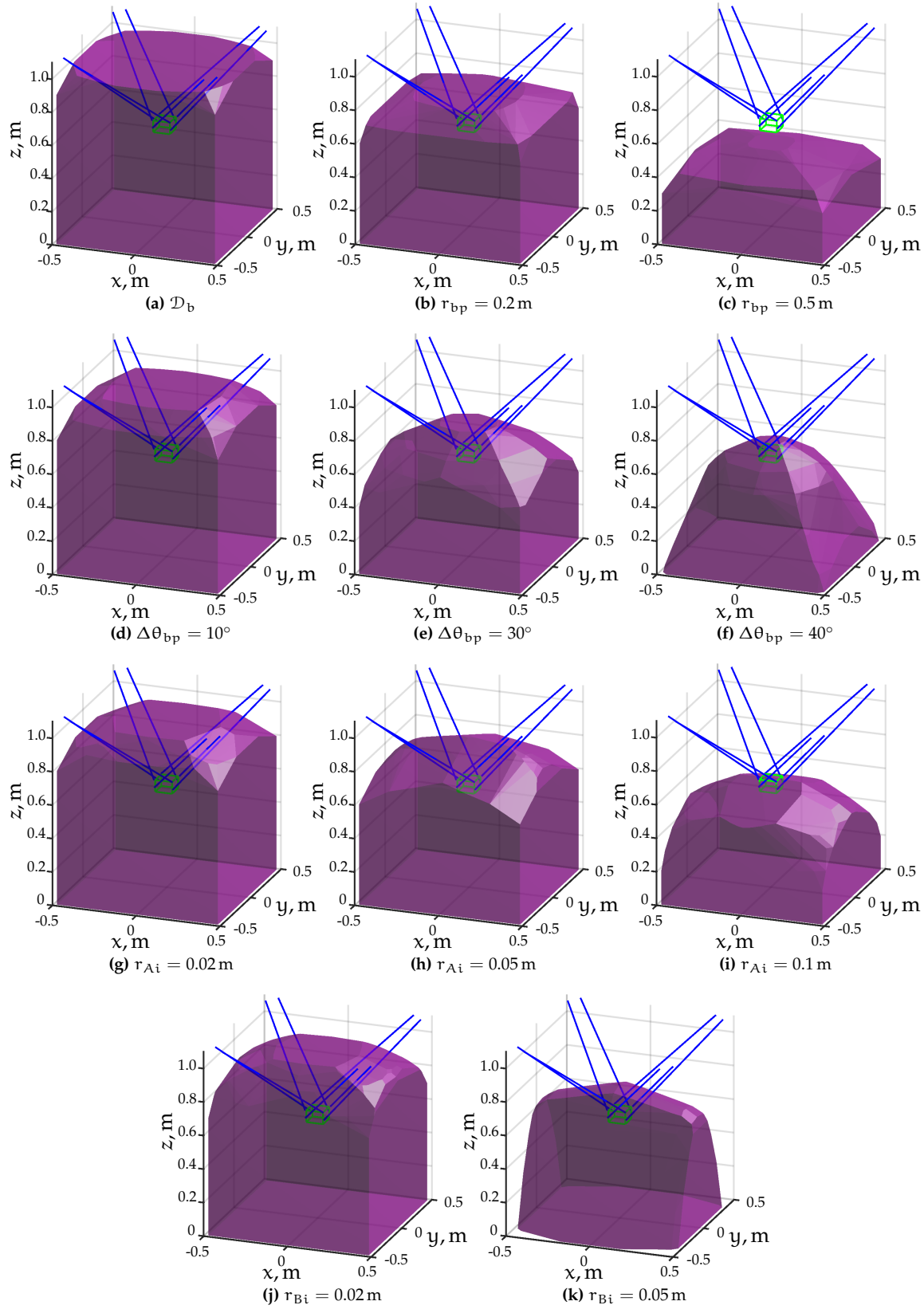


Figure 3.25: Visualization of CSW for ACROBOT with large moving-platform and IBVS

CSW for CAROCA with IBVS

The results of CSW computation for CAROCA with IBVS control are presented in this section. The ratio of CSW volume for a given perturbation set over the full volume of 82.33 m^3 is given in Fig. 3.26, while the full results can be found in Table A.18. Moreover, CSW volume visualization is shown in Fig. 3.27.

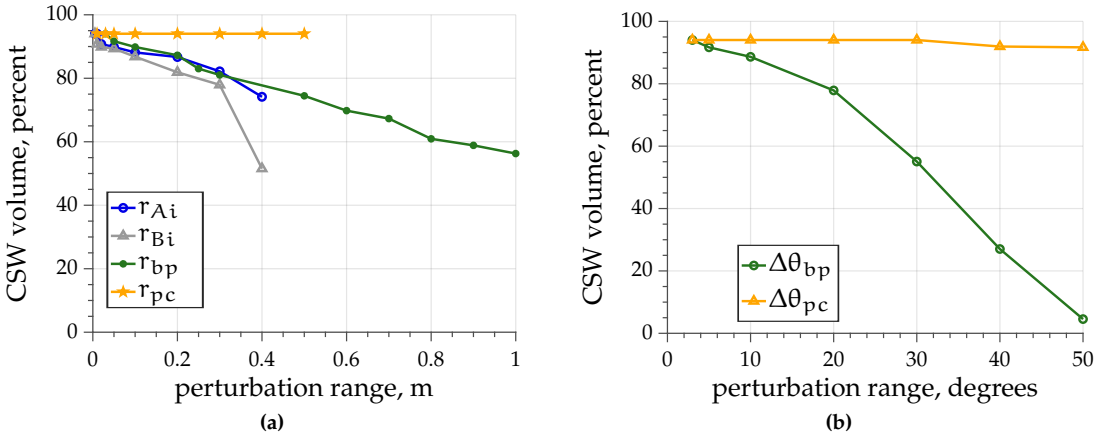


Figure 3.26: The effect of increasing perturbation range on CSW volume for CAROCA with IBVS

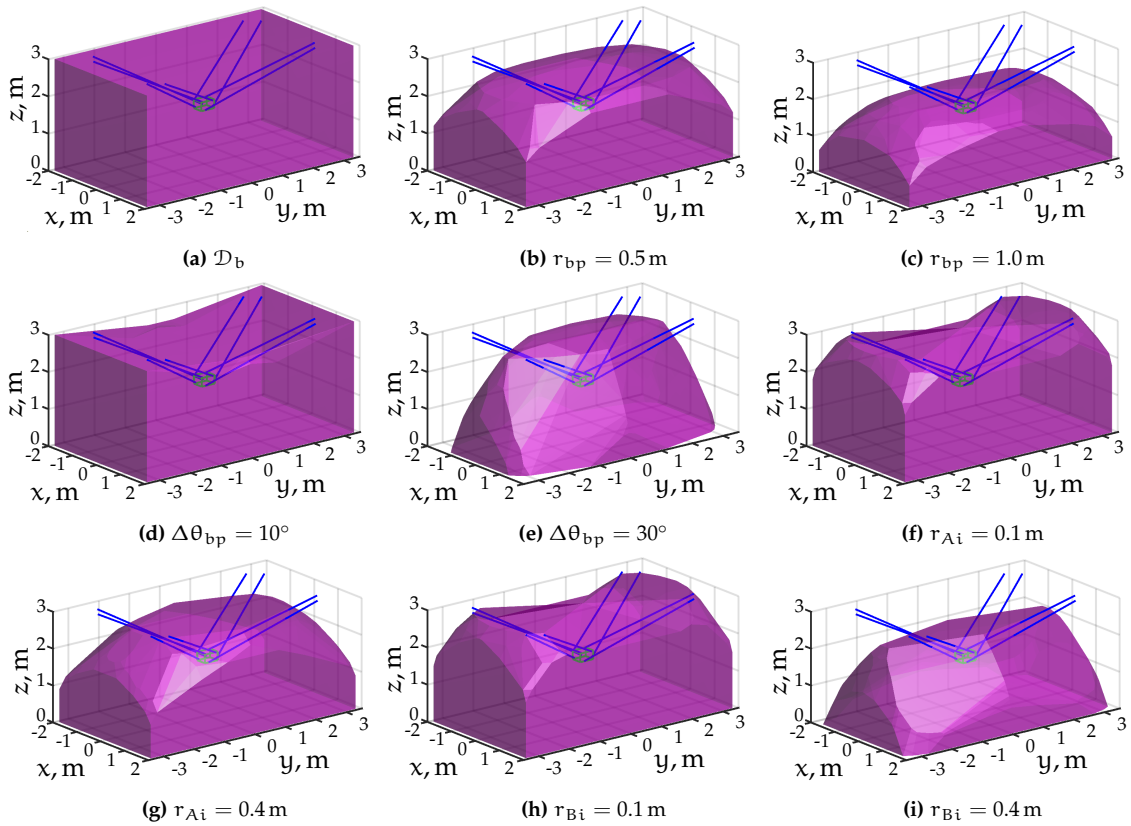


Figure 3.27: Visualization of CSW for CAROCA with IBVS

Given that for IBVS we can only compute the local asymptotic stability, it is no surprise that the resulting CSW is large. None of the tested perturbation ranges result in a void

workspace. Indeed, only with $\Delta\theta_{bp} = 50°$ does the workspace become tiny (but not void), its volume being 3.75 m^3. Even with unreasonably large perturbation values, such as $r_{Bi} = 0.4$ m (Fig. 3.27i) or $r_{bp} = 1.0$ m (Fig. 3.27c) the CSW remains above 50% of the full workspace. Note that for IBVS not only r_{pc} has no effect on the workspace, but also $\Delta\theta_{pc}$. In fact, with $\Delta\theta_{pc} \leqslant 30°$ the workspace remains at its maximum volume of 77.414 m^3. By increasing $\Delta\theta_{pc}$ up until 50° the workspace volume becomes 75.479 m^3, thus decreasing by less than 2 m^3.

Despite the increased CSW volume, the trend for IBVS is just as it was shown for PBVS and 2½D VS. More precisely:

- Perturbation on camera position in the moving-platform frame has little to no effect on CSW volume and system stability (the only exception being with 2½D VS on CAROCA);
- Similarly, perturbation on camera orientation in the moving-platform frame has the smallest effect on CSW volume, when compared to all of the remaining perturbations;
- There can be significant perturbation on the moving-platform position in the base frame without making the system unstable;
- Perturbation on the CDPR model, that is on cable exit and anchor point coordinates, has the largest effect on CSW volume;
- Perturbation on cable anchor points leads to a smaller CSW than same size perturbation on cable exit points;
- Translational perturbation bounds scale with CDPR geometry:
  - ◇ The larger the moving-platform, the larger r_{Bi} can be without making the workspace empty;
  - ◇ The larger the CDPR, the larger r_{Ai} and r_{bp} without making the workspace empty;
- On the other hand, the rotational perturbations $\Delta\theta_{bp}$ and $\Delta\theta_{pc}$ do not scale with CDPR geometry. They remain approximately the same for ACROBOT and CAROCA.

3.4 Experimental Validation of Stability Analysis and CSW

All the different experiments used to validate stability analysis and CSW computation are assembled in this section. Note that all case studies are numbered sequentially in this thesis and the first one can be found in Section 2.3.

3.4.1 Case Study II: Experimental Validation of Stability Analysis on ACROBOT with PBVS

Experimental Setup

As the very first controller, it was chosen to implement the pose-based visual servoing (PBVS) controller presented in Section 1.4.4. More precisely, the feature vector $\mathbf{s}$ is defined as $\mathbf{s} = (\mathbf{s}_t, \mathbf{s}_\omega)$, where $\mathbf{s}_t = {}^c\mathbf{t}_o$ is the translational distance between the camera

frame and the object frame; and $\mathbf{s}_\omega = \theta\mathbf{u}$ is the axis-angle representation of the rotational matrix ${}^{c^*}\mathbf{R}_c$ between the current and desired camera frames. The control scheme is shown in Fig. 1.22.

The small moving-platform of ACROBOT is equipped with the Media-tech camera. It is mounted in the eye-in-hand configuration facing the ground. AprilTags are still used as object, as presented in Chapter 2.

Adaptive gain λ, defined in (1.31), is used and the coefficients have been tuned at the following values: $\lambda_0 = 1.7$, $\lambda_\infty = 0.2$ and $\dot{\lambda} = 30$. Furthermore, to avoid velocity discontinuity at the beginning of the task, a continuous behavior is used as defined in (1.35).

To consider the task completed, the error $\boldsymbol{e}$ needs to become sufficiently small. For these experiments we define two thresholds: $k_{et} = 0.0005\,\text{m}$ and $k_{e\omega} = 0.01\,\text{rad}$. The task is considered completed if $\|\boldsymbol{e}_t\|_2 \leqslant k_{et}$ and $\theta \leqslant k_{e\omega}$, where θ is the angle from $\boldsymbol{e}_\omega$.

For ACROBOT, there are multiple sources of errors or perturbations:

i. Cable anchor points B_i cannot be precisely measured due to the mechanical connections between the cables and moving-platform. In the model B_i is considered as a point, while in experiments it is actually a point located on a ⌀0.01 m sphere around its nominal location;
ii. ACROBOT pulleys are not taken into account in the CDPR model;
iii. Moving-platform pose is estimated by control integration as shown in Section 2.2.1. The initial moving-platform pose was measured by hand, the expected accuracy is $\pm 2\,\text{cm}$.
iv. The auto-focus option of the chosen camera leads to the image being zoomed in and out. This is an additional perturbation on the AprilTag localization, leading to a less precise computation of $\mathbf{s}$.

These perturbations are covered by the baseline perturbation bound $\mathcal{D}_b$ in the CSW computation. Evidently, they are not large enough to make the system unstable, however they are also not negligible.

Three types of experiments are done. First, without any voluntarily added perturbation. Second, adding voluntary perturbations within the bounds exhibited in Section 3.2.2. And third, adding perturbations that do not respect these bounds. The perturbation input values were taken from Cb. 18 in Fig. 3.4.

To ensure the comparability of the results, each time the initial and the desired poses are the same. Only the voluntarily added perturbations change. The initial moving-platform pose ${}^b\mathbf{p}_{p0}$ and the desired moving-platform pose ${}^b\mathbf{p}_{p^*}$ are defined as:

$$ {}^b\mathbf{p}_{p0} = \begin{bmatrix} 0.097\,\text{m} & -0.026\,\text{m} & 0.323\,\text{m} & 20^\circ & -20^\circ & 45^\circ \end{bmatrix} $$

$$^{b}\mathbf{p}_{p^*} = \begin{bmatrix} 0.35\,\mathrm{m} & 0.28\,\mathrm{m} & 0.06\,\mathrm{m} & 0^\circ & 0^\circ & 0^\circ \end{bmatrix}$$

Voluntarily added perturbation within bounds of stability: $r_{bp} = 0.1\,\mathrm{m}$ and $\Delta\theta_{bp} = 5^\circ$.
Voluntarily added perturbation out of bounds of stability: $r_{bp} = 0.2\,\mathrm{m}$ and $\Delta\theta_{bp} = 10^\circ$.

Experimental Results

The experimental results are shown in Figs. 3.28 through 3.30[4]. More precisely, the AprilTag trajectory in the image is shown in Fig. 3.28; the error $\boldsymbol{e}$ over time is shown in Fig. 3.29; and the cable velocities $\dot{\mathbf{l}}$ over time are shown in Fig. 3.30.

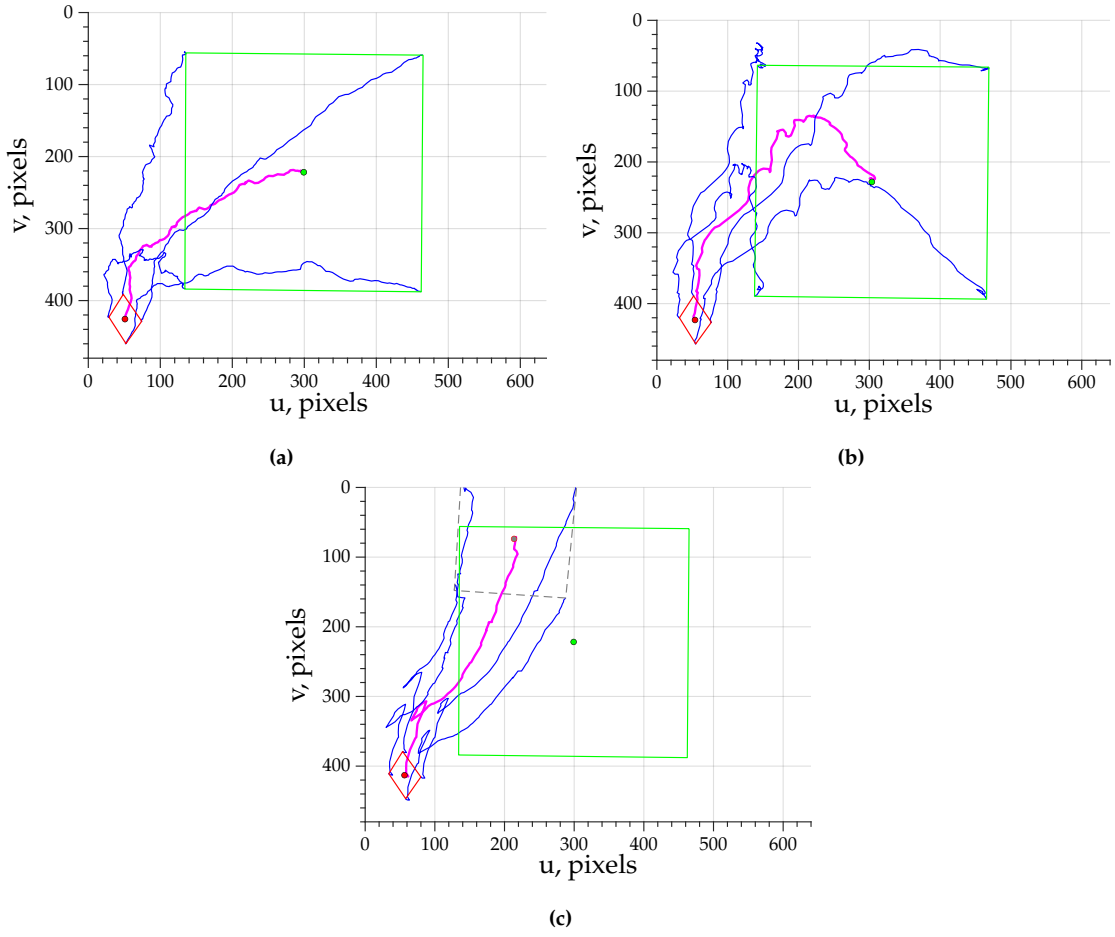


Figure 3.28: ACROBOT: AprilTag trajectory in the image: **(a)** no perturbation added; **(b)** perturbations within bounds of stability added; **(c)** perturbations out of bounds of stability are added

As can be seen in Fig. 3.28, even without added perturbation, the AprilTag trajectory in the image is not a straight-line, which is the expected outcome for an ideal system. This is due to the existing perturbations in ACROBOT, which are described above. Here, it appears that the low quality camera is the largest contributor. As will be seen in other experiments - once the MediaTech camera is replaced by the IDS camera, the behavior improves. Nevertheless, the task is succeeded both in Fig. 3.28a and in Fig. 3.28b. Indeed,

[4]Please also see the accompanying video at `https://youtu.be/gqov84Xo9Qg`

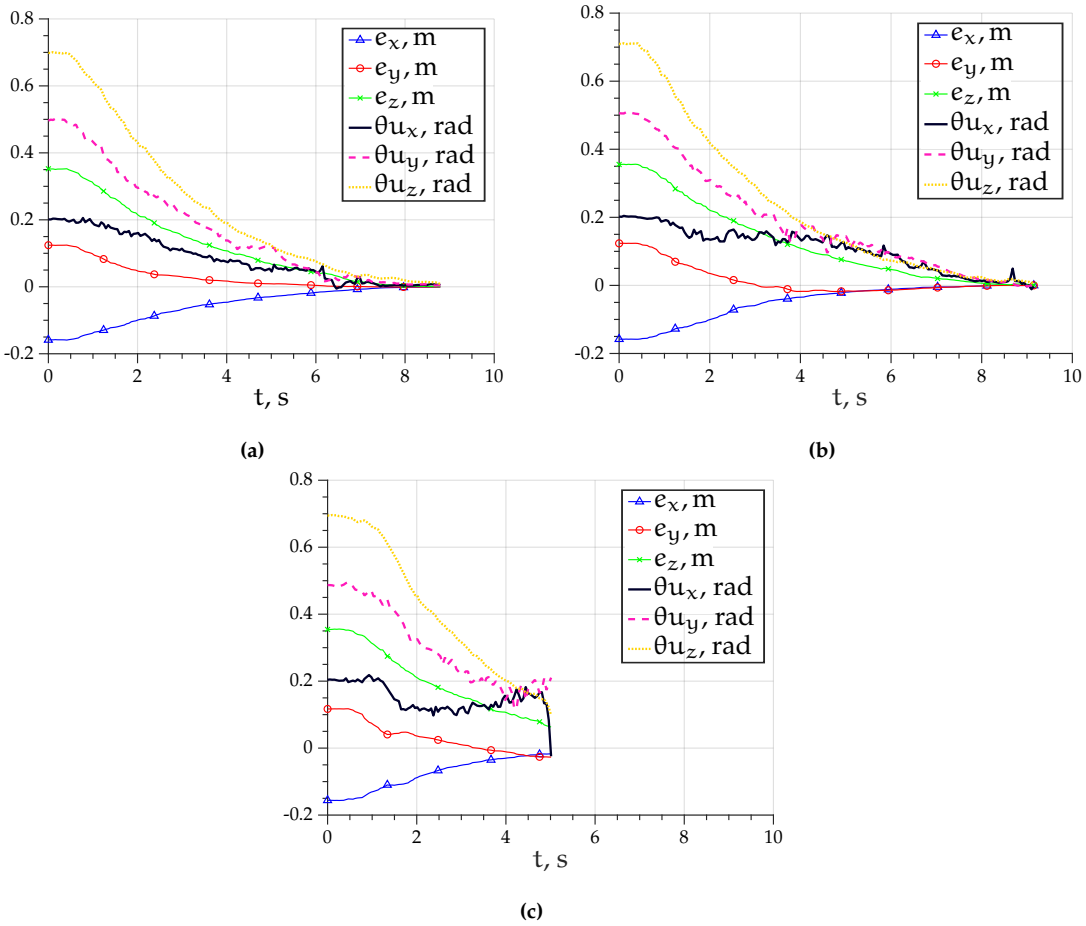


Figure 3.29: ACROBOT: Error $\boldsymbol{e}$ over time: **(a)** no perturbation added; **(b)** perturbations within bounds of stability added; **(c)** perturbations out of bounds of stability are added

as expected, the robot is able to finish its task despite having a rather coarse estimation of its initial pose. On the other hand, when the perturbation on the initial moving-platform pose is doubled, the task fails as can be seen in Fig. 3.28c. The deviation of the trajectory due to added perturbation is too large and the AprilTag leaves the image, which makes the continuation of the task impossible.

Despite the noise present in the system, a good decoupled decrease of $\boldsymbol{e}$ can be observed in Fig. 3.29a. In case of perturbations within system stability, more time is necessary for $\boldsymbol{e}$ to converge to $\mathbf{0}$, as can be seen in Fig. 3.29b.

Note that in all three experiments initially the error reduces very slowly (Fig. 3.29). This is because a continuous behavior, defined in (1.35), is used, to ensure a smooth beginning of the motion. Indeed, in Fig. 3.30 in all three experiments the velocity of each cable has a smooth transition from zero to the maximum speed.

To compare the vision-based control method of CDPRs to other existing methods, it was chosen to implement the simple position controller presented in Appendix A.4. Given the previously mentioned initial and desired moving-platform poses, ${}^{b}\mathbf{p}_{p0}$ and

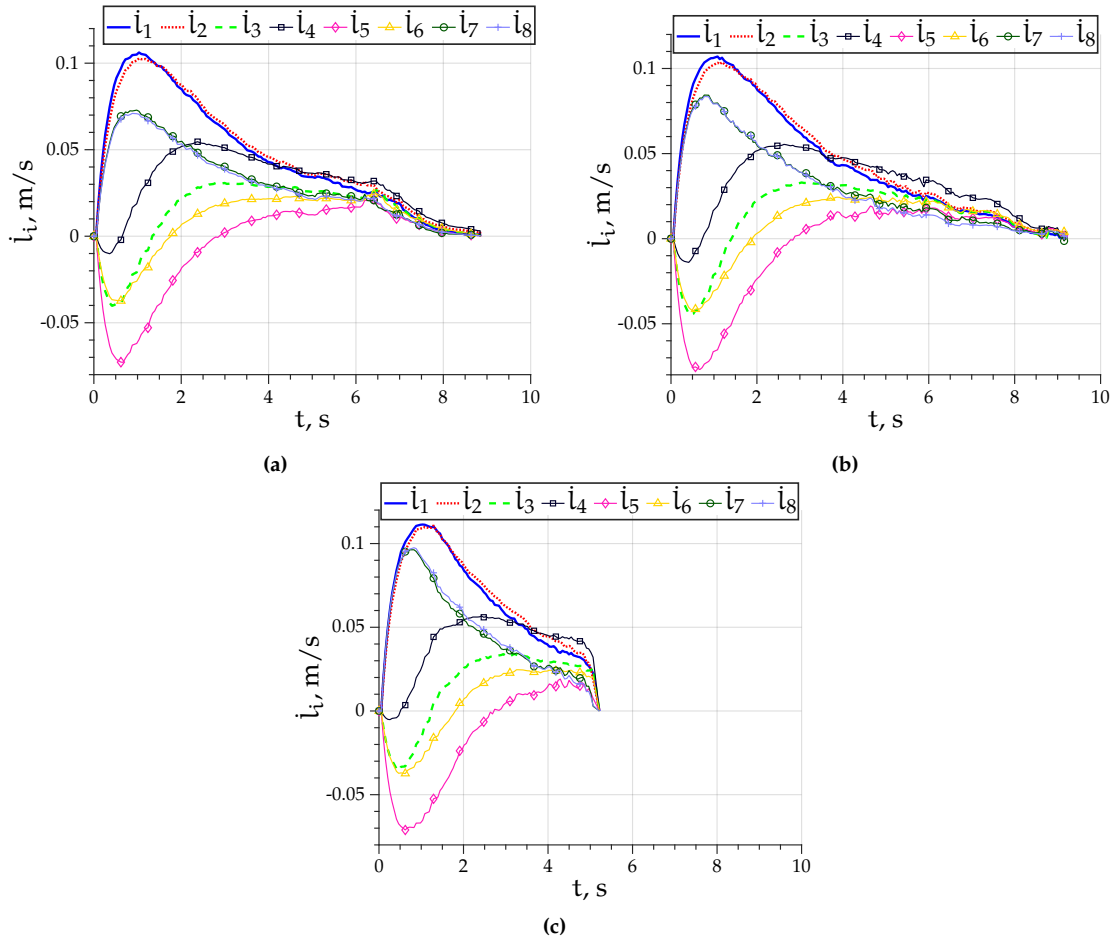


Figure 3.30: ACROBOT: Cable velocities over time: **(a)** no perturbation added; **(b)** perturbations within bounds of stability added; **(c)** perturbations out of bounds of stability are added

${}^{b}\mathbf{p}_{p^*}$ respectively, a trajectory is generated and at each iteration a command is sent to ACROBOT.

Figure 3.31 shows the final pose of the moving-platform. It appears that no matter the additional perturbations, the moving-platform reaches the same pose with PBVS control as seen in Figs. 3.31a and 3.31c. On the contrary, using position control the final moving-platform pose is far from the targeted one for both experiments shown in Fig. 3.31b and Fig. 3.31d. Indeed, the position controller does not use any external sensors, thus cannot correct its trajectory to arrive at the desired pose. Furthermore, it appears that even the small errors present in the model are significant enough to influence the final pose in Fig. 3.31b. Therefore, CDPR position control is highly sensitive to any errors in the robot model and to coarse estimation of the initial moving-platform pose. On the contrary, PBVS control turns out to be much more robust to perturbations. Indeed, as long as the system is stable, any uncertainty present in the model has an effect on the transient phase, namely, the trajectory performed to reach the goal, but not on the final reached pose. This makes a clear difference in terms of accuracy between vision-based control and a pure position controller.

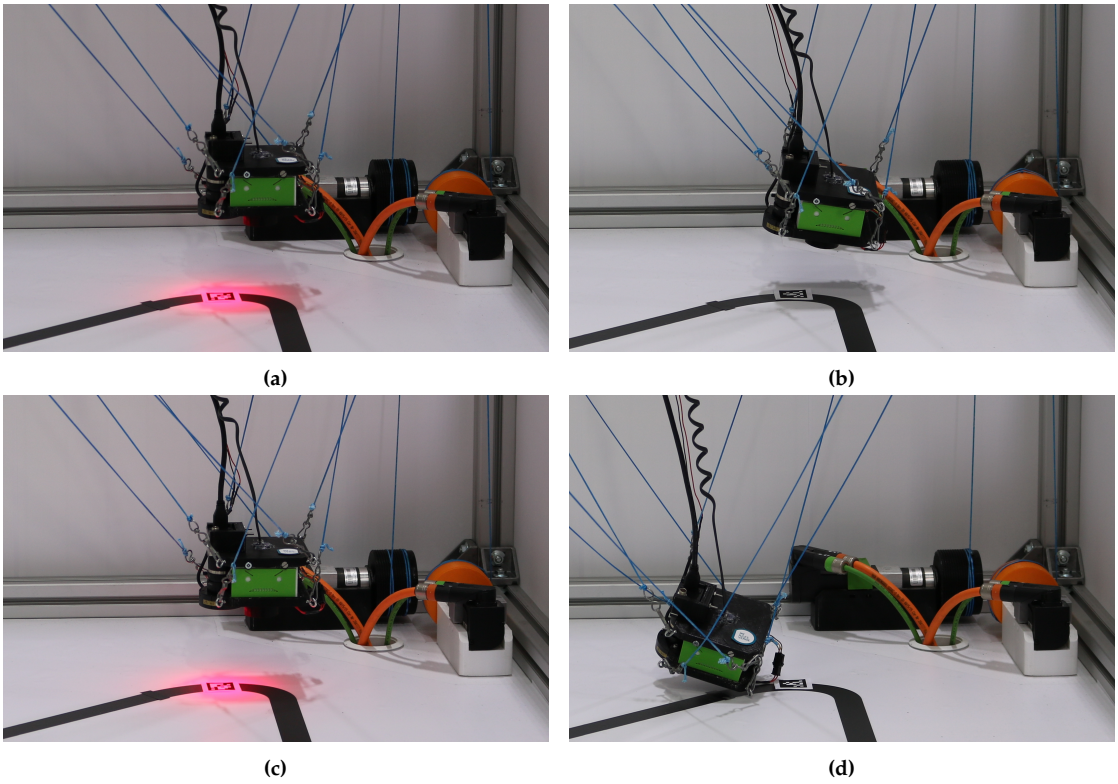


Figure 3.31: ACROBOT moving-platform pose at the end of a prescribed trajectory: **(a)** PBVS and **(b)** position control, when no perturbations are added; **(c)** PBVS and **(d)** position control, when perturbations within system stability are added

3.4.2 Case Study III: Comparison of Stability in ACROBOT and its Simulation

Clearly, the model of ACROBOT is very simple and some characteristics, such as pulley geometry, are not modeled. In addition, its geometry can have manufacturing defects and some perturbation is present in the vision system. This results in the robot not behaving as an ideal system would, as was shown in the previous experiment.

Furthermore, it was discovered in the stability analysis that all the different perturbations affect the system stability jointly. For this reason, if other perturbations exist in the system, one of them can only be increased to a strictly limited range, otherwise the CSW volume begins to rapidly reduce.

It is thus of interest to compare the difference of behavior in simulation and on a real CDPR. To do so, an additional analysis was done, where all parameters but one were assumed to be known perfectly. Then the perturbation of this single parameter was increased to find the limit at which the CSW starts to reduce in volume.

It was found that:

- No matter the translational perturbation r_{pc} value, the CSW remains at its maximum volume of $1.014\,m^3$. Indeed, even with $r_{pc} = 1.0\,m$ the workspace size does not decrease;

- The rotational perturbation $\Delta\theta_{pc}$ can be as large as 40° with the workspace remaining full, however then it starts to rapidly decrease. With $\Delta\theta_{pc} = 55°$ the workspace is 0.514 m^3 and it becomes null with $\Delta\theta_{pc} = 65°$;
- No matter the translational or rotational perturbation on feature vector $\mathbf{s}$, the workspace remains unchanged. The maximum tested values were $r_{\mathbf{s}} = 0.5$ m and $\Delta\theta_{\mathbf{s}} = 90°$;
- With cable exit point error $r_{Ai} = 0.01$ m the workspace still has a rather large volume of 0.802 m^3 and with $r_{Ai} = 0.03$ m it decreases to 0.503 m^3;
- The results for cable anchor points are very similar. With $r_{Bi} = 0.01$ m the CSW volume is 0.794 m^3 and with $r_{Bi} = 0.03$ m the CSW volume decreases to 0.506 m^3;
- With rotational error $\Delta\theta_{bp} = 3°$ of the moving-platform pose estimation, the CSW is almost at its maximum size, the volume being 0.936 m^3. It then decreases to 0.786 m^3 by increasing the error to $\Delta\theta_{bp} = 10°$;
- With translational error $r_{bp} = 0.01$ m of the moving-platform pose estimation, the workspace is also considerably large, its volume is 0.907 m^3. It decreases to 0.761 m^3 by increasing the error to $r_{bp} = 0.1$ m.

Experimental Setup

The experimental setup is almost the same as in Section 3.4.1. Only the initial and desired state has been changed to:

- initial state:
  - $\diamond$ ${}^{b}\mathbf{p}_{p0} = \left[0.097\,\text{m};\ -0.026\,\text{m};\ 0.323\,\text{m};\ 20°;\ -20°;\ 45°\right]$
  - $\diamond$ ${}^{c}\mathbf{p}_{o0} = \left[-0.02\,\text{m};\ 0.02\,\text{m};\ 0.328\,\text{m};\ 177°;\ -31°;\ 139°\right]$
- desired state:
  - $\diamond$ ${}^{b}\mathbf{p}_{p^*} = \left[0.10\,\text{m};\ 0.075\,\text{m};\ 0.09\,\text{m};\ 0°;\ 0°;\ 0°\right]$
  - $\diamond$ ${}^{c}\mathbf{p}^{*}_{o} = \left[0\,\text{m};\ 0\,\text{m};\ 0.06\,\text{m};\ 180°;\ 0°;\ 180°\right]$

Note that the moving-platform poses are shown here simply to give an idea of the motion with respect to the CDPR workspace. The desired moving-platform pose is not used in control, while the initial pose is used as the initial input for the moving-platform pose estimation.

To compare the ideal and the real CDPR, a rotational perturbation $\Delta\theta_{pc}$ is added to the system. According to the findings described above, the ideal system would still be stable with $\Delta\theta_{pc} = 55°$. As can be seen in Fig. 3.32 the corresponding CSW is quite large even though the corners of the workspace have become unavailable. When taking into account that this perturbation is not the only one present in the system, the workspace becomes null. More precisely, we used the baseline perturbation set $\mathcal{D}_b$ and increased the rotational error to $\Delta\theta_{pc} = 55°$, for which not a single moving-platform pose could be found so that $\boldsymbol{\Pi} > \mathbf{0}$. The moving-platform pose shown in Fig. 3.32 corresponds to ${}^{b}\mathbf{p}_{p0}$ and the green dot notes the position of ${}^{b}\mathbf{p}_{p^*}$. Note that the cyan dashed line is the shortest path, however with the chosen feature vector $\mathbf{s}$ definition the moving-platform

trajectory is not controlled. Indeed, it is only the object center-point trajectory in the image that should be a straight line ideally.

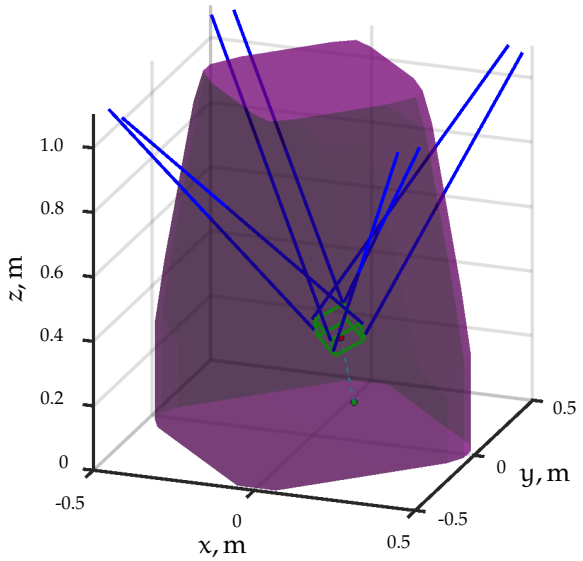
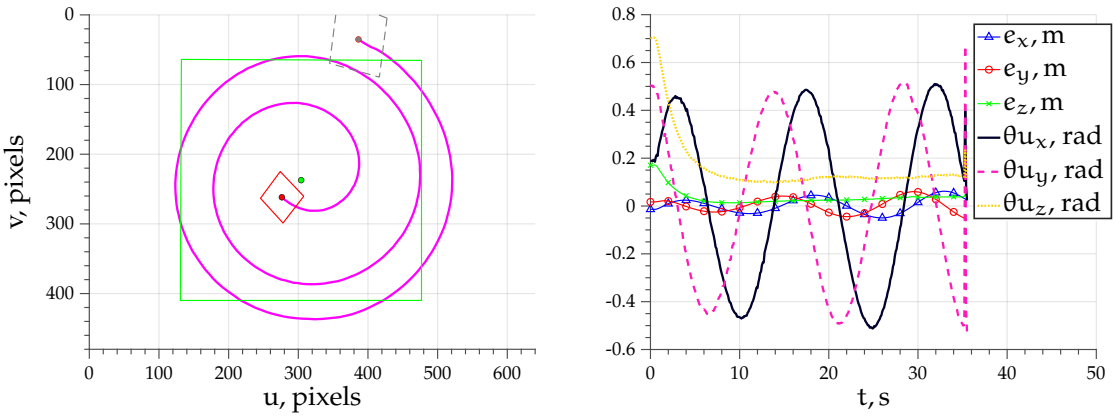


Figure 3.32: CSW for a simulated CDPR with only one perturbed parameter: $\Delta\theta_{pc} = 55°$

Experimental Validation

Here, the behavior of a CDPR prototype ACROBOT and its simulation in V-REP is compared. Note that the V-REP model is presented in Appendix A.1.3. The experimental results are shown in Figs. 3.33 to 3.35[5].



(a) **(b)**

Figure 3.33: V-REP simulation with $\Delta\theta = 85°$: **(a)** AprilTag center-point trajectory; **(b)** error $\boldsymbol{e}$ over time. System is not stable

In the first experiment, shown in Fig. 3.33, the simulated system has a large rotational error on the camera pose expressed in moving-platform frame $\mathcal{F}_p$. More precisely, $\Delta\theta_{pc} = 85°$ and the axis of this angle corresponds to the vertical z axis. As shown in Fig. 3.33a the AprilTag center-point steadily diverges from the desired position in the image center. Indeed, the motion can be described as an outward spiral and continues

[5]Please also see the accompanying video at `https://youtu.be/tfiTDlp1ZIY`

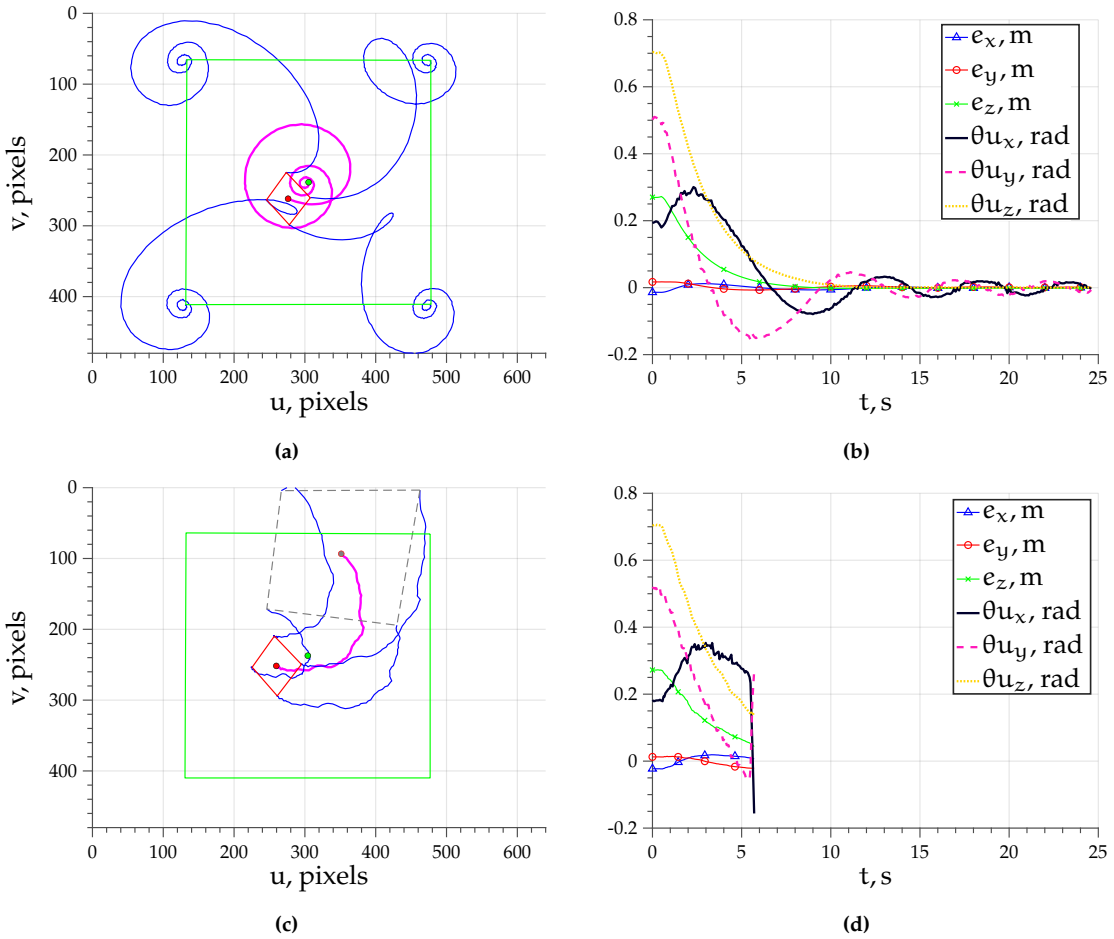


Figure 3.34: CDPR behavior depending on added perturbations. **(a)** and **(b)**: the AprilTag trajectory and error $\boldsymbol{e}$ over time in V-REP with $\Delta\theta = 55°$, system is stable; **(c)** and **(d)**: the AprilTag trajectory and error $\boldsymbol{e}$ over time on ACROBOT with $\Delta\theta = 55°$, robot does not converge because AprilTag leaves the camera field of view.

until the AprilTag leaves the field of view, which leads to a task failure. Consequently, each component of the error $\boldsymbol{e}$ oscillates with an increasing amplitude, as shown in Fig. 3.33b. Clearly, while attempting to reduce the error in order for $\mathbf{s}$ to converge to $\mathbf{s}^*$ the computed control signal actually makes the error increase. Thus, the system is not stable.

According to the results of stability analysis, if we set $\Delta\theta_{pc} = 55°$ the simulated robot should still succeed. The AprilTag trajectory is shown in Fig. 3.34a and indeed the task is finished successfully. Of course, the trajectory is highly perturbed, as is expected with such large perturbations in the system. The effect of the perturbations is also clearly visible on the errors plots in Fig. 3.34b. It can be seen that the trajectory takes a long time and some components of $\boldsymbol{e}$ oscillate around the horizontal axis with a decreasing amplitude. The same rotational error on the actual robot is shown in Fig. 3.34c. The initial behavior is similar, but the radius of the deviation is too large and the AprilTag leaves the image, leading to a task failure. This is not surprising, given that ACROBOT is not ideal and thus there are small differences between the robot model and ACROBOT.

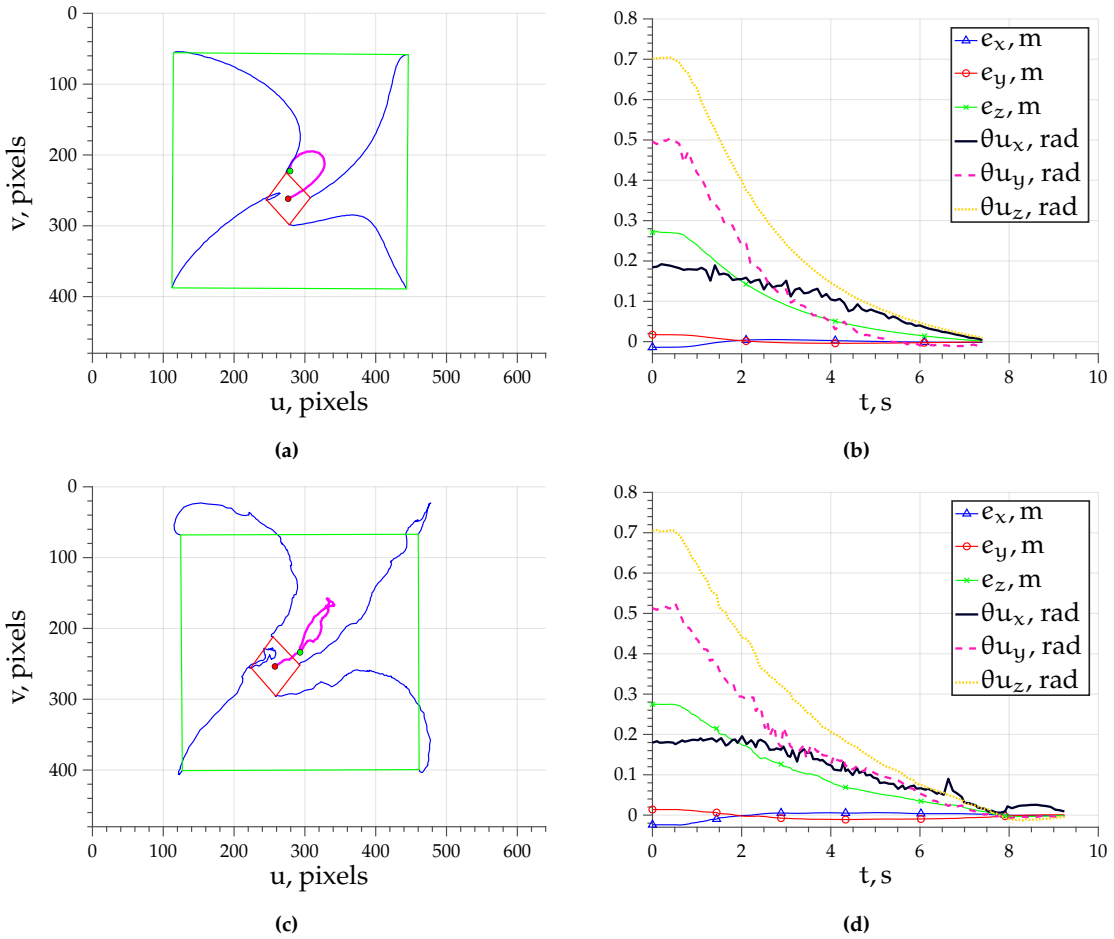


Figure 3.35: CDPR behavior depending on added perturbations. **(a)** and **(b)**: the AprilTag trajectory and error $\boldsymbol{e}$ over time in V-REP with $\Delta\theta = 16°$; **(c)** and **(d)**: the AprilTag trajectory and error $\boldsymbol{e}$ over time on ACROBOT with $\Delta\theta = 16°$; system is stable in both experiments

Finally, $\Delta\theta_{pc}$ was set to a value that is within the bounds of stability, i.e. $\Delta\theta_{pc} = 16°$. This time not only the simulated robot, but also ACROBOT finishes the task successfully. As can be seen in Fig. 3.35, AprilTag trajectory by ACROBOT is more perturbed and it takes a slightly longer time to make the error $\boldsymbol{e}$ converge to zero. Furthermore, there are no more oscillations in the error plots.

To summarize, for an ideal robot the range of perturbation on a single parameter is very large. However, a real robot is not ideal and the use of the simple kinematic model permits some small errors in the system. Once this is taken into account, then each individual perturbation has quite a limited range within the bounds of stability. Nevertheless, the control is very robust to many perturbations and their combinations.

3.4.3 Case Study IV: PBVS on CAROCA

Experimental Setup

In this case study the CAROCA prototype is used with the IDS camera mounted on its moving-platform. The initial and desired states are defined as follows:

- initial state:
  - ◇ ${}^{b}\mathbf{p}_{p0} = \left[0.713\,\text{m};\ 0.102\,\text{m};\ 1.899\,\text{m};\ -18°;\ 10°;\ -5°\right]$
  - ◇ ${}^{c}\mathbf{p}_{o0} = \left[-1.042\,\text{m};\ 0.709\,\text{m};\ 2.277\,\text{m};\ -159°;\ -10°;\ -179°\right]$
- desired state:
  - ◇ ${}^{b}\mathbf{p}_{p^*} = \left[-0.689\,\text{m};\ -1.374\,\text{m};\ 0.757\,\text{m};\ 0°;\ 0°;\ 0°\right]$
  - ◇ ${}^{c}\mathbf{p}_{o}^{*} = \left[0\,\text{m};\ 0\,\text{m};\ 0.6\,\text{m};\ 180°;\ 0°;\ 180°\right]$

Perturbation within bounds of stability amounts to 0.2 m and 4° of error on the initial moving-platform pose. The perturbation out of bounds of stability is 0.4 m and 8°.

The adaptive gain λ for PBVS of CAROCA is tuned as in Section 2.3.

Experimental Results

The experimental results are shown in Figs. 3.36, 3.37 and 3.38.

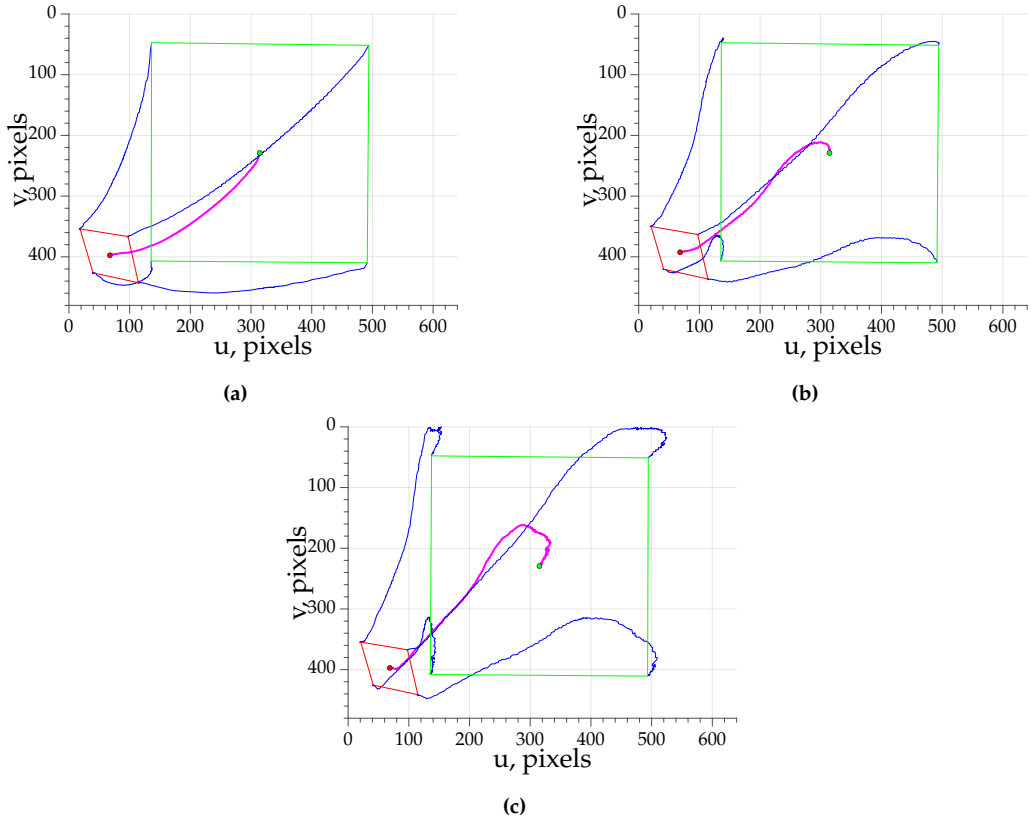


Figure 3.36: CAROCA: AprilTag trajectory in the image: **(a)** no perturbation added; **(b)** perturbations within bounds of stability added; **(c)** perturbations out of bounds of stability are added

The AprilTag trajectories, shown in Fig. 3.36 are smoother than the ones in Fig. 3.28 for ACROBOT. This is due to the change of camera. Here, an industrial camera is used and it outputs an image of higher quality, which reduces the noise of AprilTag localization. However, the trajectory in Fig. 3.36a is not a straight line. It can be explained by the simplified CDPR model that is used in the control. CAROCA has large pulleys of 0.15 m in diameter and its steel cables tend to sag, especially when the moving-platform is not

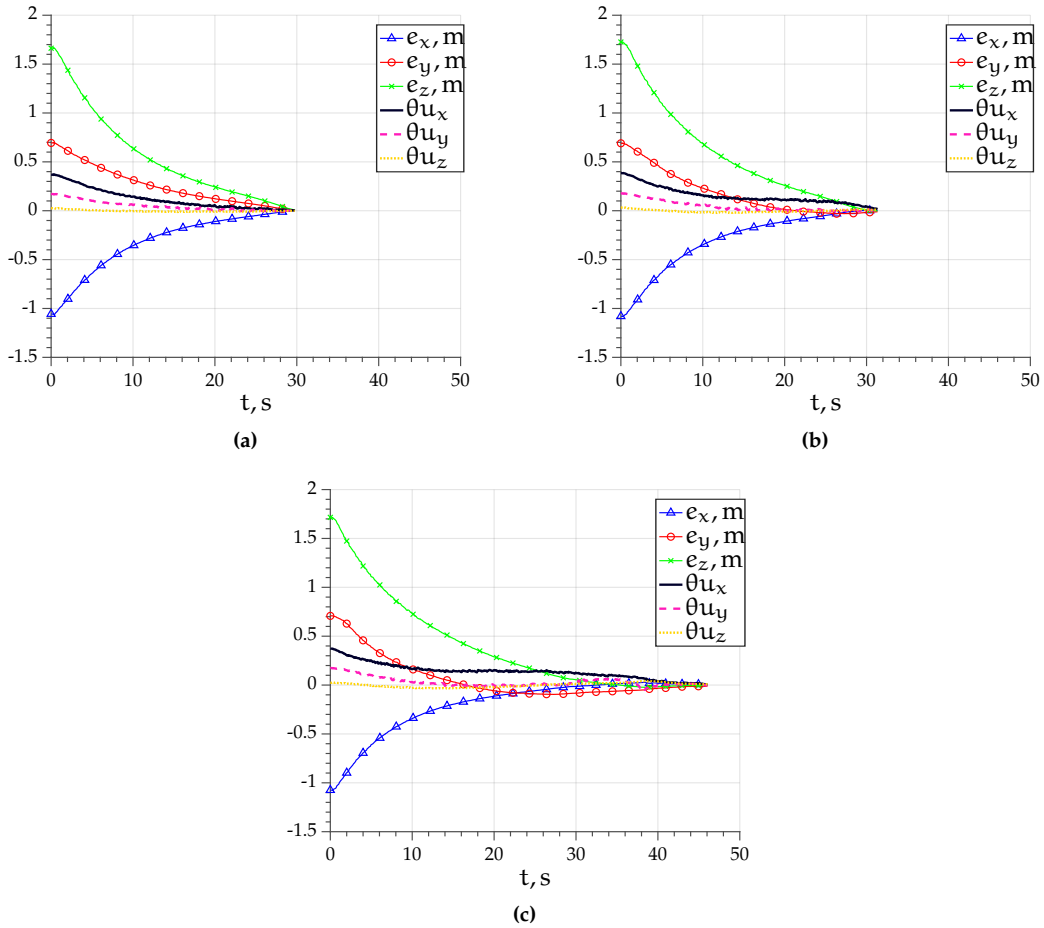


Figure 3.37: CAROCA: Error $\boldsymbol{e}$ over time: **(a)** no perturbation added; **(b)** perturbations within bounds of stability added; **(c)** perturbations out of bounds of stability are added

carrying any payload [Pic18]. Nonetheless, the deviation is rather small and as can be seen in Fig. 3.37a the error $\boldsymbol{e}$ reduces smoothly. Similarly, there are no abrupt changes in cable velocity curves in Fig. 3.38a.

Adding a 0.2 m and 4° perturbation on the initial moving-platform pose only slightly affects the trajectory of AprilTag center-point shown in Fig. 3.36b. As a reminder, a 0.2 m perturbation caused task failure on ACROBOT, as shown in Fig. 3.28c. For CAROCA the error $\boldsymbol{e}$ reduces almost identically in Fig. 3.37b as in Fig. 3.37b, where the former takes only 2 s more time to finish the task.

When the large perturbation is applied, the AprilTag almost leaves the image. Note the two upper corners of the AprilTag in Fig. 3.36c, where they pass by the very margin of the image frame. However in the end the task is successfully completed despite the perturbation being out of bounds of stability. Indeed, criterion $\boldsymbol{\Pi} > \mathbf{0}$ is only a sufficient condition and when it is not held the stability of the system is unknown. The initial motion produced by the controller lead to decreasing the distance to the desired state. As mentioned before, the smaller the error $\boldsymbol{e}$ the more robust the system to different

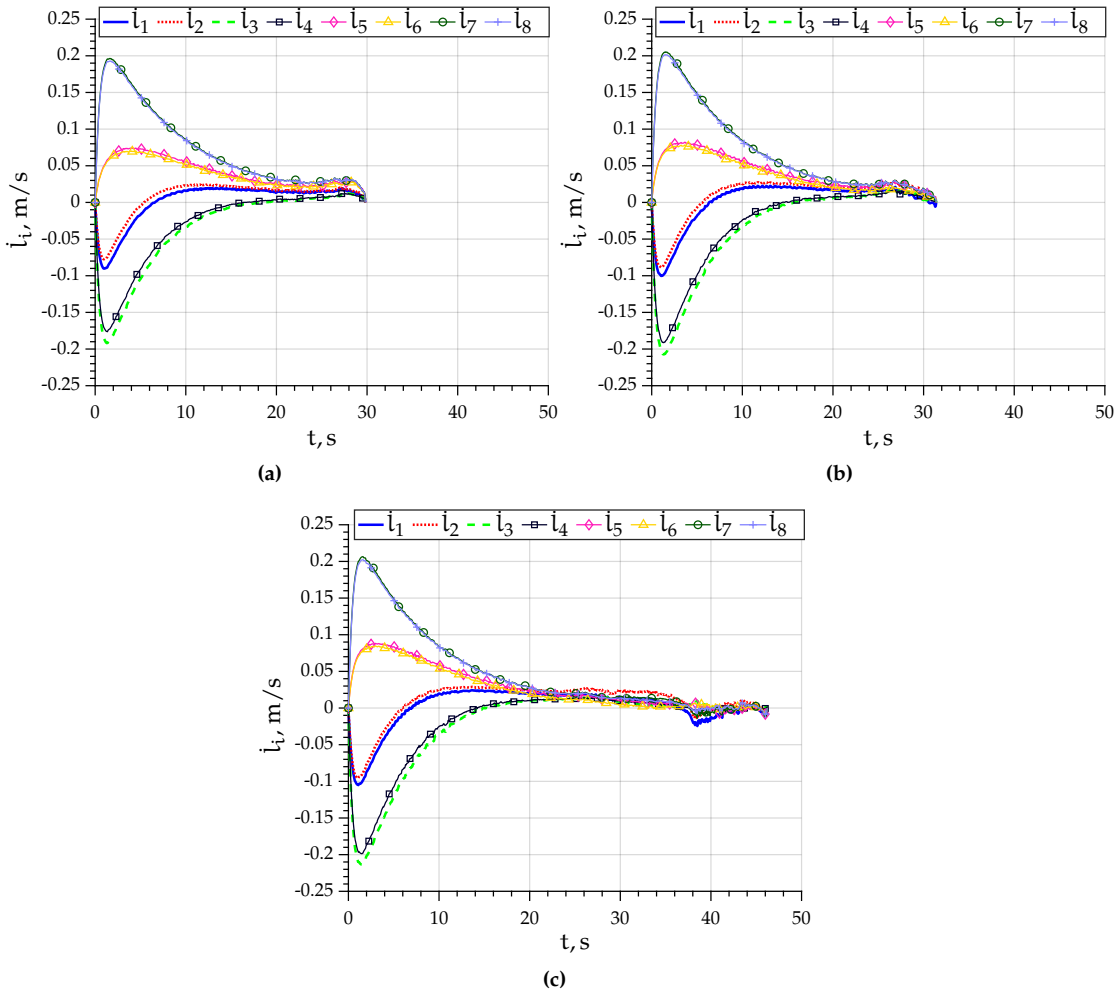


Figure 3.38: CAROCA: Cable velocities over time: **(a)** no perturbation added; **(b)** perturbations within bounds of stability added; **(c)** perturbations out of bounds of stability are added

perturbation. Thus along the trajectory the stability criterion is fulfilled again and the task is successfully finished. Note that this trajectory took about 46 s, which is 50% more than for the other two experiments. Furthermore, some of $\boldsymbol{e}$ components, such as e_y change their sign over time. Note also the abrupt increase of cable velocities at $t = 38\,s$ in Fig. 3.38c, which is an undesired behavior. This increase of velocities corresponds to θu_y changing its sign to negative, as can be seen in Fig. 3.37c.

It can be concluded that the perturbation bounds within system stability depend on the CDPR size. Indeed, for a larger CDPR the perturbation can also be larger without causing task failure. Furthermore, even though pulleys were not modeled, the CDPR successfully finished its task with the same accuracy despite the different perturbations. On the other hand, not taking pulleys into account leads to a non-ideal trajectory being produced.

3.4.4 Case Study V: 2½D VS on CAROCA

As shown in previous sections, PBVS of CDPRs is robust to perturbations and very accurate. However, due to the selected feature vector, the camera trajectory to the goal is not controlled. Indeed, only the target center-point trajectory in the image is a straight line. By choosing 2½D VS and defining the feature vector as shown in Section 1.4.5 both the target center-point trajectory in the image and the camera Cartesian trajectory should be straight lines.

The control strategy for 2½D VS is the same as for PBVS, shown in Fig. 1.22. It should be kept in mind that the composition of the feature vector $\mathbf{s}$ (subsequently its desired value $\mathbf{s}^*$ and error $\boldsymbol{e}$) and the interaction matrix $\mathbf{L_s}$ is different from PBVS, as defined in Section 1.4.5.

Experimental Setup

The experiment is defined as in Section 3.4.3, including the initial and desired moving-platform poses in base frame $\mathcal{F}_b$ and object pose in camera $\mathcal{F}_c$. 2½D VS, requires to also define the AprilTag center-point coordinates in the image, which are the following:

- initial state:
  - $\mathbf{o}_0 = \left[-0.469\,\text{m};\ 0.304\,\text{m}\right]$
- desired state:
  - $\mathbf{o}^* = \left[0.0\,\text{m};\ 0.0\,\text{m}\right]$

Results

The results are shown in Fig. 3.39. Without added perturbation the camera trajectory, shown in Fig. 3.39c in blue appears almost ideal. Indeed, the deviation from the straight-line trajectory is at most 7 cm as can be seen in Fig. 3.39d. It a very small deviation when compared to the traveled distance of about 2.3 m. The AprilTag trajectory, shown in Fig. 3.39a is more perturbed. Note that this deviation is smaller for 2½D VS than for PBVS, which is shown in Fig. 3.36a. Thus, without added perturbation using 2½D VS we obtain a slightly better object trajectory in the image as well as a controlled camera trajectory.

Once the 0.2 m and 4° error on the initial moving-platform pose is added, the behavior of the controller becomes perturbed. The image trajectory, shown in Fig. 3.39a in pink, is only slightly worse at the beginning, however this kind of perturbation induces a large overshoot at the end of the trajectory. It is even visible for the camera trajectory shown in Fig. 3.39c in pink, where at the very end the curve surpasses the desired state that is indicated by a green dot. By doubling the error in the initial moving-platform pose the task fails because the two upper corners of AprilTag leave the image during the overshoot. Furthermore, doubling the perturbation produces twice as large deviation of camera trajectory, as can be seen by comparing pink and orange curves in Figs. 3.39c and 3.39d.

Exactly the same experimental setup was used for 2½D VS as for PBVS in Section 3.4.3. The resulting behavior however is different. Indeed, the produced AprilTag center-point

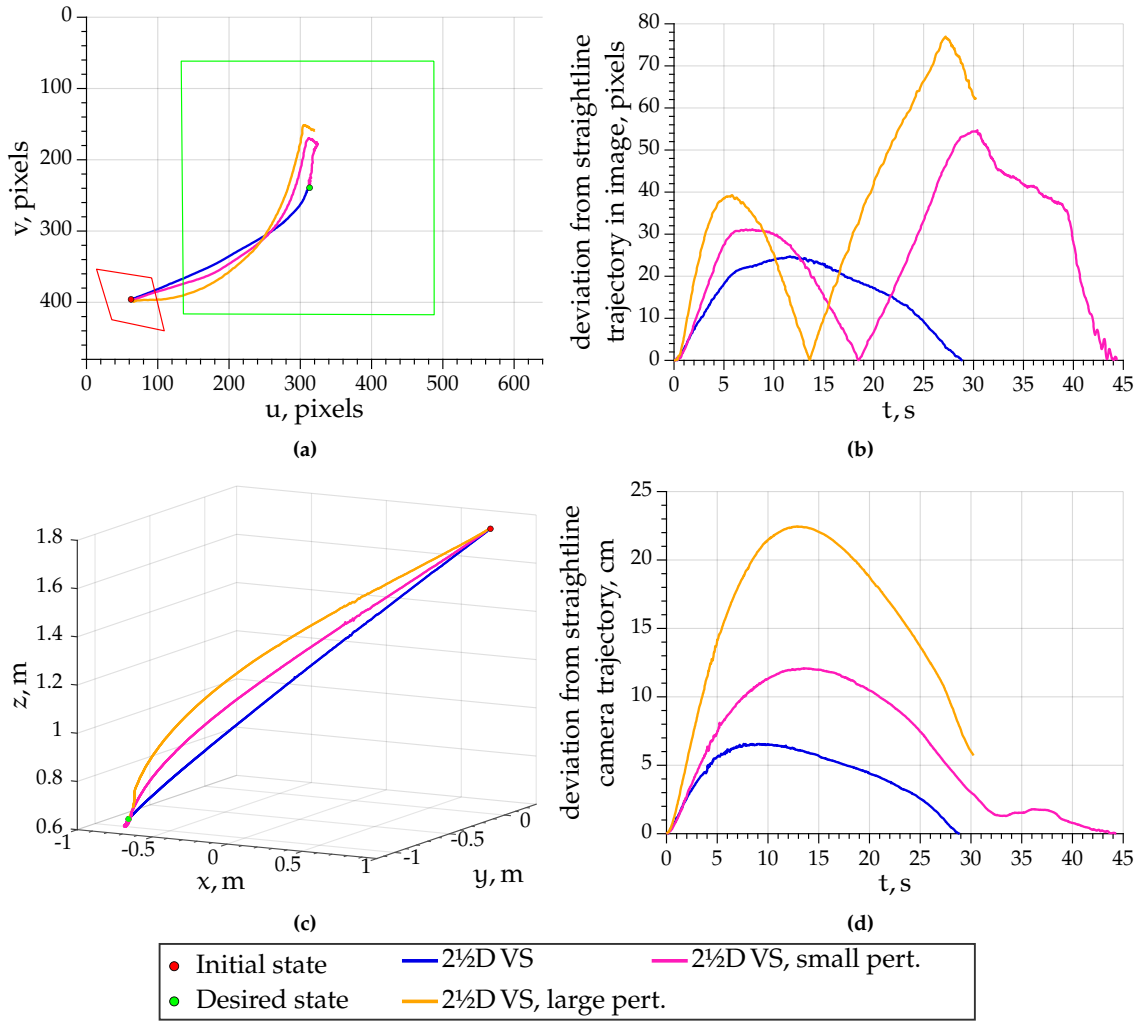


Figure 3.39: 2½D VS experiments on CAROCA: **(a)** the trajectory of AprilTag center-point in the image; **(b)** The pixel deviation from the ideal straight-line trajectory; **(c)** the trajectory of the camera in the frame $\mathcal{F}_b$; **(d)** the deviation from the ideal straight-line 3D trajectory.

trajectory in the image was more curved for PBVS without any added perturbation, while with supplementary perturbation in the system 2½D VS produced large overshoot that lead to task failure.

3.4.5 Case Study VI: IBVS on ACROBOT

Experimental Setup

In this section IBVS control is used on ACROBOT with the IDS camera mounted on its moving-platform. A pattern of four points, shown in Fig. 3.40, is used as an object. The control scheme is shown in Fig. 1.22.

Adaptive gain λ, defined in (1.31), is used and the coefficients have been tuned at the following values: $\lambda_0 = 1.0$, $\lambda_\infty = 0.2$ and $\dot{\lambda} = 30$.

The initial and the desired states are defined as follows:

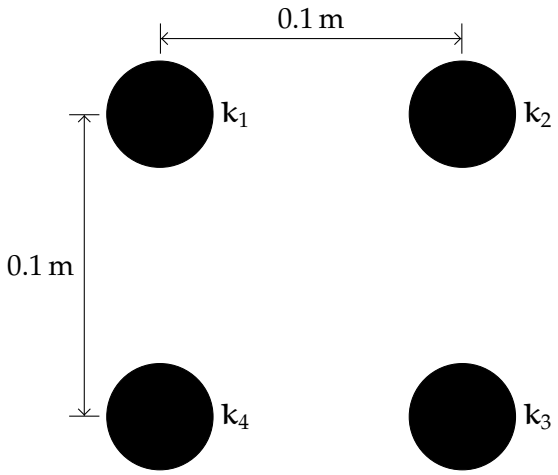


Figure 3.40: Pattern with four points used for IBVS

- initial state:
  - ◇ ${}^{\mathrm{b}}\mathbf{p}_{\mathrm{p}_0} = \left[0.118\,\mathrm{m};\ -0.104\,\mathrm{m};\ 0.459\,\mathrm{m};\ 16^\circ;\ 13.8^\circ;\ 20.6^\circ\right]$
  - ◇ $\mathbf{k}_1 = \left[-0.276;\ -0.281\right]$
  - ◇ $\mathbf{k}_2 = \left[-0.116;\ -0.213\right]$
  - ◇ $\mathbf{k}_3 = \left[-0.179;\ -0.040\right]$
  - ◇ $\mathbf{k}_4 = \left[-0.329;\ -0.114\right]$
- desired state:
  - ◇ ${}^{\mathrm{b}}\mathbf{p}_{\mathrm{p}^*} = \left[0.314\,\mathrm{m};\ -0.006\,\mathrm{m};\ 0.262\,\mathrm{m};\ 0^\circ;\ 0^\circ;\ 0^\circ\right]$
  - ◇ $\mathbf{k}_1^* = \left[-0.167;\ -0.167\right]$
  - ◇ $\mathbf{k}_2^* = \left[0.167;\ -0.167\right]$
  - ◇ $\mathbf{k}_3^* = \left[0.167;\ 0.167\right]$
  - ◇ $\mathbf{k}_4^* = \left[-0.167;\ 0.167\right]$

where $\mathbf{k}_1$ to $\mathbf{k}_4$ are the normalized coordinates of the four points in the image that are used in the feature vector $\mathbf{s}$ in this IBVS experiment.

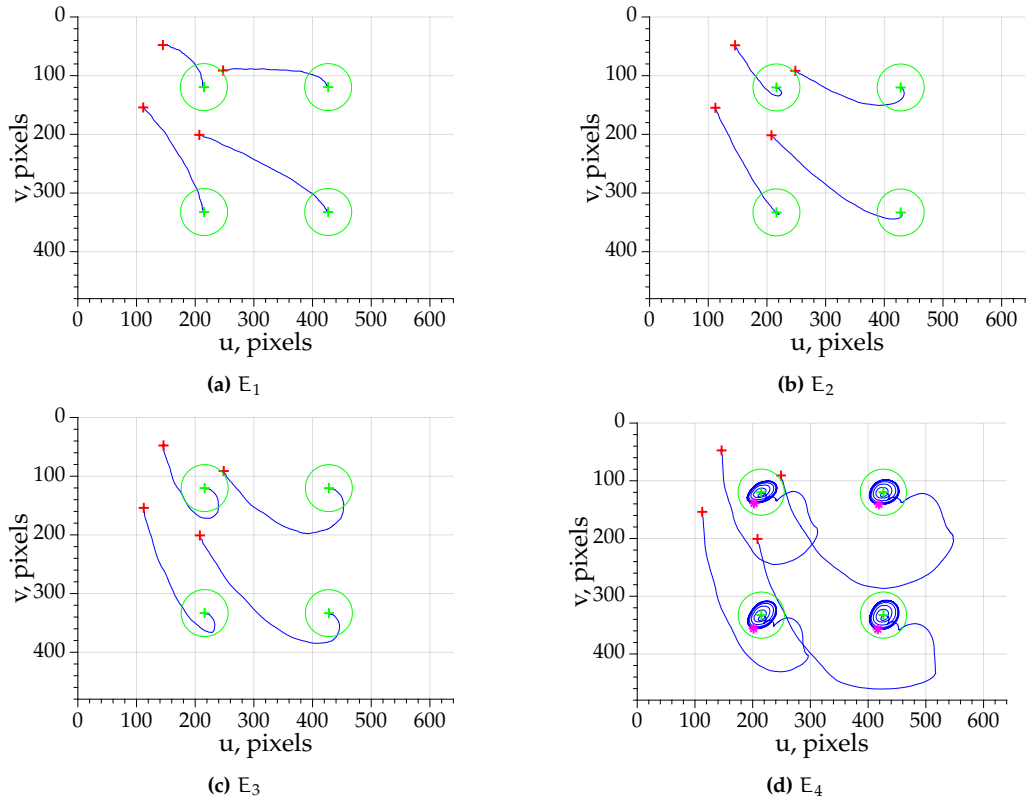
Several experiments are done with the following perturbations:

- E_1 - no perturbation added;
- E_2 - $r_{bp} = 0.08\,\mathrm{m}$ and $\Delta\theta_{bp} = 10^\circ$;
- E_3 - $r_{bp} = 0.145\,\mathrm{m}$ and $\Delta\theta_{bp} = 19^\circ$;
- E_4 - $r_{bp} = 0.28\,\mathrm{m}$ and $\Delta\theta_{bp} = 37^\circ$.

Please note that additional experiments with the same experimental setup and different perturbations on different parameters are shown in Appendix A.6.

Experimental Validation

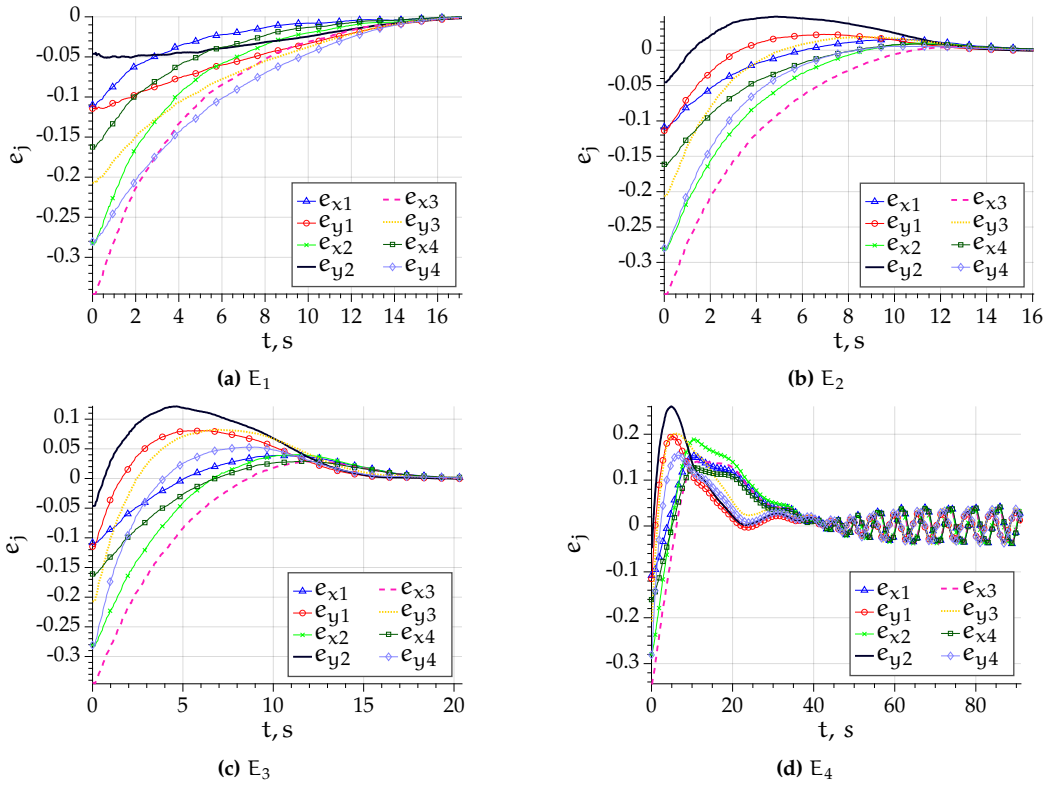
The experimental results are shown in Figs. 3.41 through 3.44. The initial position of the four points in the image is shown with red crosses in Fig. 3.41, while the desired one is shown with green crosses. Note that in experiment E_4 the final position does not correspond to the desired one and is shown with pink stars in Fig. 3.41d.



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

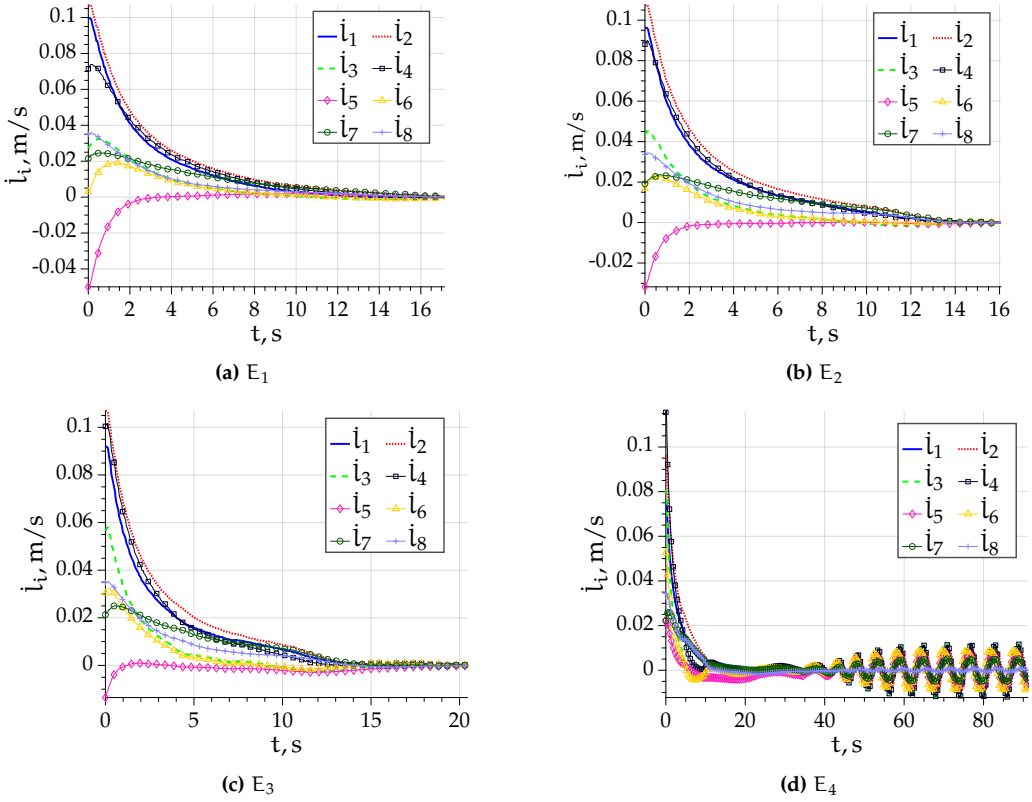
Figure 3.41: Trajectory of four points in the image



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

Figure 3.42: Error e over time



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

Figure 3.43: Cable velocities over time

In experiment E_1 the trajectories of the four points are almost straight lines, as can be seen in Fig. 3.41a. Furthermore, the components of error $\boldsymbol{e}$, shown in Fig. 3.42a, rapidly converge to zero. The trajectory of the moving-platform is shown in Fig. 3.44 in blue and it is smooth, with no deviation.

A small perturbation is added in E_2, which leads to a curved trajectory of the four points in the image, as shown in Fig. 3.41b. This corresponds to the overshoot of error $\boldsymbol{e}$ components in Fig. 3.42b. For example, the black curve of e_{y2} begins at $-0.045\,m$ and reaches $0.045\,m$ at $t = 5\,s$ before starting to converge to zero. The trajectory of the moving-platform shown in pink in Fig. 3.44 is deviated.

The perturbation is doubled in E_3. Here, the point trajectories have become significantly more perturbed in Fig. 3.41c, however the task is still finished successfully. The overshoot in the error plots in Fig. 3.42c has also increased. For e_{y2} it now reaches $0.12\,m$ before converging to zero. Note that the convergence to zero takes about $4\,s$ longer in E_3. The trajectory of the moving-platform shown in orange in Fig. 3.44 is even more deviated than in E_2.

The perturbation is doubled once more in E_4. While for all of the previous experiments the task was finished successfully, in E_4 that is not the case. The trajectory to the goal is highly perturbed and in the vicinity of the goal the system becomes unstable. Indeed, as can be seen in Fig. 3.42d, the errors had almost converged to zero at $t = 40\,s$, however

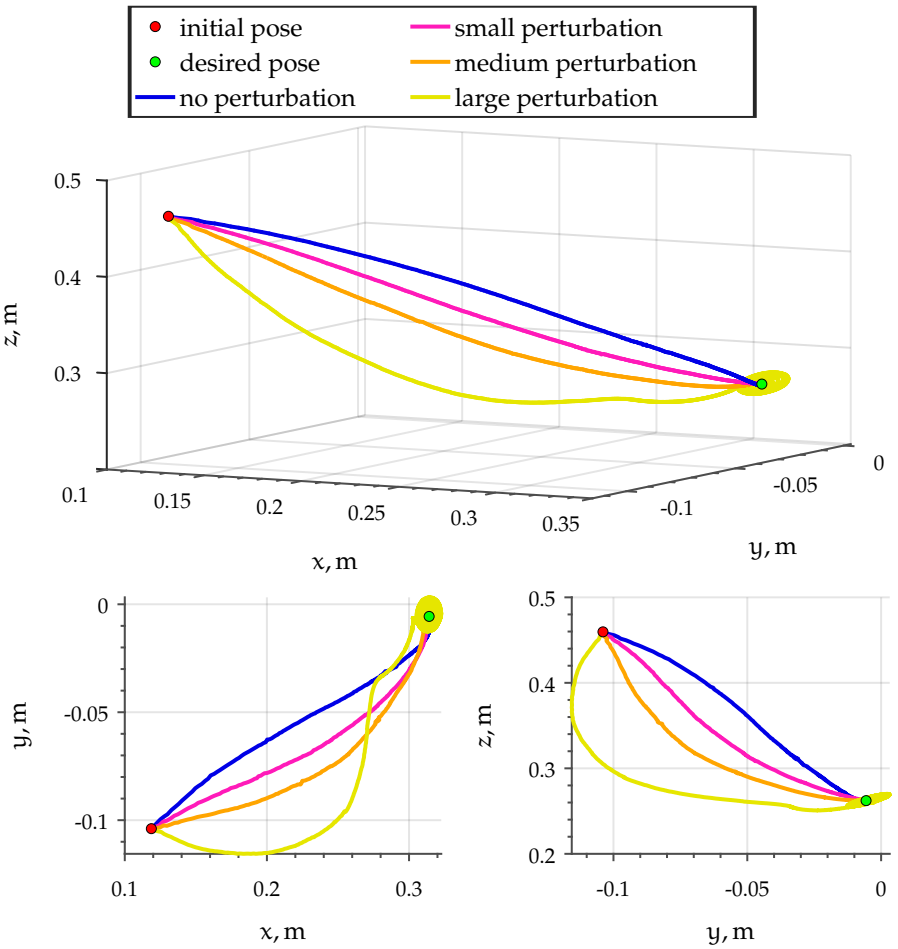


Figure 3.44: Moving-platform trajectory

instead the error oscillates around zero with an increasing amplitude. Note that at about $t = 65\,s$ the amplitude of the oscillation is no longer increasing. The controller appears to be stuck in the vicinity of the desired state unable to converge. The same oscillations can also be seen in cable velocity plots in Fig. 3.43d. Finally, the outward spiral that can be seen in the image trajectory in Fig. 3.41d, can also be seen in the moving-platform trajectory shown in Fig. 3.44. The yellow curve for E_4 is highly deviated from other curves. Near the desired pose a spiral can be seen. Here, however the changes in the moving-platform pose were too close and the trajectory curve looks like a filled disk at the end.

Thus, similarly as for PBVS and 2½D VS, IBVS is a robust visual servo control that allows us to increase the accuracy of the CDPR at the desired state. Furthermore, this is true even though only the local asymptotic stability of the system can be demonstrated, as described in Section 3.2.3. Despite the added perturbations, the accuracy at the desired state is the same, as long as the system is stable. However, when the system becomes unstable, the behavior can be unpredictable. In this case getting stuck close to the desired state and oscillating around it endlessly.

The additional experiments, shown in Appendix A.6, allow us to evaluate the system behavior when a different parameter is perturbed. It can be concluded that having only a coarse estimation of the camera position on the moving-platform does not greatly affect the robot behavior. Indeed, changing the estimated position to three different arbitrary points leads only a slight perturbation of the produced trajectories. On the contrary, perturbation on cable anchor points has a significant effect on robot behavior and the produced trajectories.

3.5 Conclusions

After initial proof of concept study, it was concluded that visual servoing control of CDPRs is possible and is robust to some uncertainties and perturbations.

To quantify the robustness of the system with respect to perturbations, stability analysis was done. It was shown that many different perturbations can exist, including modeling errors. However, it was also shown that the system is highly robust to perturbations, despite their quantity and combined effect on the stability.

From the analysis carried out in Section 3.2.2, it turns out that a CDPR with PBVS controller is little sensitive to uncertainties in the robot model. This is of course true for all the visual servoing schemes. Since the object of interest is observed with a visual servoing control, it is possible to know whether the desired pose has been reached or not. It means that no matter the amount of perturbations in the system, the moving-platform will always reach its targeted pose as long as the system is stable.

With a visual servoing control, the CDPR accuracy mainly depends on camera quality and the task completion threshold. Indeed, the task completion threshold is in fact the goal accuracy. If it is set too coarse, then the accuracy is also coarse. On the other hand, setting the threshold to be very small can lead to a task that cannot be finished, because the measurement noise is larger than the threshold.

When analyzing the separate perturbation's effect on stability, it was found that for an ideal system any one parameter could be highly perturbed without making the system unstable. This result was successfully validated in simulation in V-REP. Surely, when attempting to control ACROBOT with same large perturbation, the system does not converge. However, if the applied perturbation is kept within the corresponding (noisy system) range, the robot will be able to complete its task. Thus, it is clear that there is a large difference between a simulated and a real CDPR, which can be the reason why model-based control approaches are struggling with accuracy.

For ACROBOT, the tolerated perturbation on cable exit points is rather large. This is beneficial, because it allows us to avoid adding pulley kinematics to the CDPR model. Indeed, cable exit point variations on the pulley are smaller than the tolerated perturbation, which does not affect the stability of the system.

It was discovered that the stability of the system depends on the moving-platform pose. Indeed, using the same perturbation set the stability criterion can be positive definite for one moving-platform pose and negative for another. For this reason a novel workspace, named Control Stability Workspace (CSW), was defined and computed for ACROBOT and CAROCA with each of the three different visual servoing controllers. Having a workspace computed for different perturbation sets allows us to evaluate the effect of each perturbation on the workspace size. Indeed, depending on the perturbed parameter, the decrease of the CSW volume happens at different rate. For example, translational error of camera pose in the moving-platform frame appears to have almost no effect on the CSW volume. On the contrary, CDPR model parameters such as cable exit and anchor points cannot be highly perturbed. With a 0.1 m error for ACROBOT on the cable anchor points renders the CSW void no matter the chosen visual servoing approach. On the other hand, such a large error on each of the cable exit of anchor points for such a small CDPR is rather unlikely. Furthermore, with reasonably sized perturbations, such as 0.005 m to 0.01 m, the CSW remains large at no less than 75% of the full workspace size for PBVS and above 80% for 2½D VS and IBVS.

The perturbation range that does not make the system unstable depends on the CDPR size. Indeed, having a 0.2 m translational error on the initial moving-platform pose caused task failure on ACROBOT, while only slightly affected task execution on CAROCA. Furthermore, on CAROCA the task was successful even with 0.4 m translational error. Similarly, the CSW remains significantly larger for increased cable exit and anchor point coordinate errors. For example, for CAROCA with the above mentioned perturbation $r_{Bi} = 0.1$ m, the CSW remains at 48.8% for PBVS and above 80% for 2½D VS and IBVS. Indeed, while not modeling the pulleys on CAROCA has some effect on the produced trajectory, it does not make the system unstable. This is because the pulley radius is 0.06 m and when we set $r_{Ai} = 0.06$ m the resulting CSW of PBVS is at about 60% of the full workspace, easily covering the trajectories of our experiments. Note that it is even larger for 2½D VS and IBVS.

Indeed, the CSW volume of the three visual servoing approaches differs. For example, the same kind of small to moderate sized perturbation results in a larger workspace for 2½D VS than for PBVS. This is especially visible for the large CDPR CAROCA. On the other hand, once the perturbation increases, the CSW of 2½D VS reduces very rapidly. Regarding IBVS, the workspace is always larger than for the other two visual servoing approaches, no matter the perturbation. However, it should be noted that for IBVS only the local asymptotic stability can be evaluated and thus the computed CSW is more like an illustration.

Finally, using an onboard camera to observe an object of interest is beneficial in order to increase robot accuracy. Indeed, as the camera moves together with the moving-platform, it also approaches the object of interest. Furthermore, the smaller the error between the current and the desired state, the better the visual servoing behavior. This characteristic is

used in Chapter 4 to improve the behavior of the robot in the presence of perturbations by using trajectory planning and tracking and thus having a time-varying desired state.

An onboard camera cannot observe the moving-platform itself, thus the well known issue of the knowledge of the moving-platform pose arises. In these experiments the moving-platform pose is estimated by control integration, which of course is an open-loop estimation and thus it is bound to drift. Similarly, the cables are not observed either and can become slack. This is even more true because of the moving-platform pose estimation method and the fact that many experiments include voluntarily perturbing the system. While the system appears stable from a control point of view, it can become underactuated due to cable slack. This issue is addressed in Chapter 5.

4. Trajectory Planning and Tracking used to Improve Robustness

Contents

4.1 Introduction

It was shown in the previous chapter that visual servoing of a CDPR is highly robust. However, even if perturbation levels are kept within the boundaries of stability, they have an undesirable effect along the trajectory to the goal.

To further improve the robustness and the ability to achieve the expected trajectory, planning and tracking of a trajectory can be used. Trajectory planning and tracking take advantage of stability and robustness to large perturbations of classical visual servoing approaches in the vicinity of the goal [MC02]. Indeed, when the difference between current and desired visual features is small, the behavior of the system approaches the ideal one, even in the presence of large perturbations as long as the system is stable. With the implementation of trajectory planning and tracking, the desired features are varying along the planned trajectory keeping the difference between current and desired visual features small at all times.

4.2 Definition of Trajectory Planning and Tracking Algorithm

Once trajectory planning and tracking is considered, the model of the system is written from Eqs. (1.18), (1.59) and (1.36):

$$\dot{\boldsymbol{e}} = \mathbf{L_s}\,\mathbf{A}_d^{-1}\mathbf{A}^{\dagger}\,\dot{\mathbf{l}} - \dot{\mathbf{s}}^* \tag{4.1}$$

Injecting (1.30), (1.36) and (1.59) into (1.18) and expressing cable velocity vector $\dot{\mathbf{l}}$ leads to :

$$\dot{\mathbf{l}} = \widehat{\mathbf{A}}\,\widehat{\mathbf{A}}_d\,\widehat{\mathbf{L}}_s^{-1}(-\lambda\boldsymbol{e} + \widehat{\dot{\mathbf{s}}}^*) \tag{4.2}$$

Hence, the new control scheme is shown in Fig. 4.1.

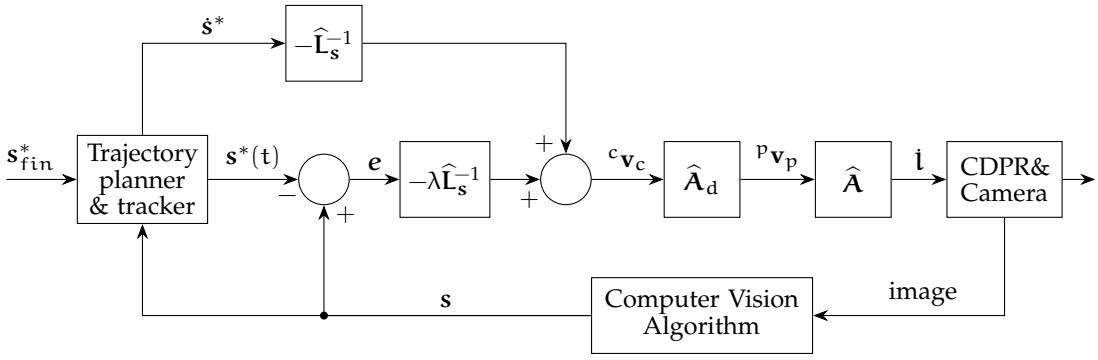


Figure 4.1: Control scheme for VS with trajectory tracking of a CDPR

The implementation of the trajectory planning and tracking for 2½D VS is shown in Algorithm 1. The strategy is similar for PBVS and IBVS.

Algorithm 1: Trajectory planning and tracking

Initialization
  Set the desired object pose ${}^{c^*}\mathbf{p}^*_{o_{fin}}$ and center-point coordinates $\mathbf{o}^*_{fin}$
  Define final feature vector $\mathbf{s}^*_{fin}$
  Read and record initial object pose ${}^{c}\mathbf{p}_{o_{init}}$ and center-point coordinates $\mathbf{o}_{init}$
  Define initial feature vector $\mathbf{s}_{init}$
  Compute trajectory time t_{full} from (4.4)
  Compute the constant velocity $\boldsymbol{\nu}$ as in (4.5)
End of Initialization

Trajectory Planning
  $\mathbf{s}^*(0) = \mathbf{s}_{init}$
  $c_f = t_{full}/\Delta t$
  for $c = 1 : c_f$ **record**
    $\mathbf{s}^*(c\Delta t) = \mathbf{s}_{init} + c\Delta t\,\boldsymbol{\nu}$
  end for
End of Trajectory Planning

Trajectory Tracking
  while $\left\|\mathbf{s}(t) - \mathbf{s}^*_{fin}\right\|_2 >$ threshold **do**
    Retrieve current desired feature vector $\mathbf{s}^*(t)$
    Compute current feature vector $\mathbf{s}(t)$
    Compute current error $\boldsymbol{e}(t) = \mathbf{s}(t) - \mathbf{s}^*(t)$
    Compute current $\widehat{\mathbf{L}}_s$, $\widehat{\mathbf{A}}$ and $\widehat{\mathbf{A}}_d$
    Compute $\dot{\mathbf{l}}$ using (4.2) and send to CDPR
  end while
End of Trajectory Tracking

There are three distinct phases: (i) initialization; (ii) trajectory planning; (iii) trajectory tracking. During the initialization phase, the final desired object pose ${}^{c^*}\mathbf{p}^*_{o_{fin}}$ and center-point $\mathbf{o}^*_{fin}$ are defined. They are used to compute the final feature vector $\mathbf{s}^*_{fin}$. Similarly, the initial feature vector $\mathbf{s}_{init}$ is defined based on the initial pose ${}^{c}\mathbf{p}_{o_{init}}$ and center-point $\mathbf{o}_{init}$ of the object of interest that are measured and recorded. This allows us to compute the full error:

$$\boldsymbol{e}_{full} = \mathbf{s}_{init} - \mathbf{s}^*_{fin} \tag{4.3}$$

and trajectory time:

$$t_{full} = \max\left(\frac{e_h}{v_h}, h = 1 \text{ to } 6\right) \tag{4.4}$$

where e_h stands for the h-th component of $\boldsymbol{e}_{full}$; v_h stands for the h-th component of the desired average velocity $\mathbf{v}$.

The current desired feature vector $\mathbf{s}^*(t)$ varies at a constant velocity $\boldsymbol{\nu}$ that is expressed as:

$$\boldsymbol{\nu} = \frac{\boldsymbol{e}_{full}}{t_{full}} \tag{4.5}$$

At the trajectory planning phase $\mathbf{s}^*(t)$ is defined. At the beginning, when $t = 0\,s$, it is clear that $\mathbf{s}^*(0) = \mathbf{s}_{init}$. Then for $c = 1, \ldots, c_f$, where $c_f = t_{full}/\Delta t$ and Δt is the time interval between two iterations, the trajectory planning is expressed as:

$$\mathbf{s}^*(c\Delta t) = \mathbf{s}_{init} + c\,\Delta t\,\boldsymbol{\nu} \tag{4.6}$$

As a consequence, we can set in (4.2):

$$\widehat{\dot{\mathbf{s}}}^* = \dot{\mathbf{s}}^* = \begin{cases} \boldsymbol{\nu} & \text{when} \quad t < t_{full} \\ \mathbf{0} & \text{when} \quad t \geqslant t_{full} \end{cases} \tag{4.7}$$

The third phase iterates until the difference $\left\|\mathbf{s}(t) - \mathbf{s}^*_{fin}\right\|_2$ reaches a defined threshold. At each iteration, the current feature vector $\mathbf{s}(t)$ is computed from the current object pose and the current object center-point coordinates. The current desired feature vector $\mathbf{s}^*(t)$ is retrieved from trajectory planning algorithm. This allows us to compute the current error $\boldsymbol{e}(t) = \mathbf{s}(t) - \mathbf{s}^*(t)$, which is then used as input of the control scheme.

When trajectory tracking is involved, the closed-loop equation is written by injecting (4.2) into (4.1). Then, by using (4.7), we obtain:

$$\dot{\boldsymbol{e}} = -\lambda\,\mathbf{L_s}\,\mathbf{A}_d^{-1}\mathbf{A}^\dagger\widehat{\mathbf{A}}\,\widehat{\mathbf{A}}_d\widehat{\mathbf{L}}_s^{-1}\boldsymbol{e} + \mathbf{L_s}\,\mathbf{A}_d^{-1}\mathbf{A}^\dagger\widehat{\mathbf{A}}\,\widehat{\mathbf{A}}_d\widehat{\mathbf{L}}_s^{-1}\dot{\mathbf{s}}^* - \dot{\mathbf{s}}^* \tag{4.8}$$

The stability criterion $\boldsymbol{\Pi}$ keeps the form defined in (3.1). However, even if $\boldsymbol{\Pi}$ is positive definite, the error $\boldsymbol{e}$ will decrease to zero if and only if the estimations are sufficiently accurate so that

$$\mathbf{L_s}\,\mathbf{A}_d^{-1}\mathbf{A}^\dagger\widehat{\mathbf{A}}\,\widehat{\mathbf{A}}_d\widehat{\mathbf{L}}_s^{-1}\dot{\mathbf{s}}^* \approx \dot{\mathbf{s}}^* \tag{4.9}$$

Otherwise tracking errors will be observed. Indeed, VS with trajectory tracking is in this way similar to VS for tracking a moving object, described in Section 1.4.6.

Most importantly, as the current desired feature vector $\mathbf{s}^*(t)$ approaches regularly the final desired feature vector $\mathbf{s}^*$, the desired feature velocity vector $\dot{\mathbf{s}}^*_{fin}$ will become $\mathbf{0}$ as stated in (4.7), which makes the tracking errors vanish at the end.

4.3 Case Study VII: Trajectory Planning and Tracking on ACROBOT

4.3.1 Experimental Setup

The small CDPR prototype ACROBOT is used for this experimental validation. The IDS camera is mounted on the small moving-platform facing the ground. AprilTags are still used as objects.

In order to pick the initial and desired moving-platform poses for this experiment, the workspace must first be computed. The presence of modeling and manufacturing errors on cable exit and anchor points are taken into account and they are assumed to be up to 0.005 m along any arbitrary axis. Similarly, hand-eye calibration errors in camera pose in the moving-platform frame $\mathcal{F}_p$ are simulated as 0.01 m along and 3° about any arbitrary axis. This corresponds to the baseline set of perturbation bounds defined as $\mathcal{D}_b$ in Section 3.3.3 and to the workspace shown in Fig. 3.20a. In order to verify the controller behavior in presence of large perturbations, we compute new workspaces for increased perturbations on the initial moving-platform pose, the camera pose on the moving-platform and cable anchor and exit points. Furthermore, we are interested in changing the orientation of the moving-platform, thus the computed CSW allows for up to $\pm 30°$ rotation of the moving-platform about any arbitrary axis. The new workspaces are shown in Fig. 4.2. The workspaces are noticeably smaller due to the increased perturbations.

Four perturbation sets are defined as $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$ and $\mathbf{d}_4$. These sets are within the bounds $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$ shown in Figs. 4.2a, 4.2b, 4.2c, and 4.2d, respectively. The set $\mathbf{d}_1$ includes the following perturbations:

- a perturbation of initial moving-platform pose of 0.19 m and 8.4° about axis $\mathbf{u} = [0.73;\ 0.67;\ -0.14]$, making the assumed initial moving-platform pose ${}^{b}\widehat{\mathbf{p}}_{p0} = [0\,\text{m};\ -0.15\,\text{m};\ 0.25\,\text{m};\ 15°;\ -25°;\ 3°]$;
- a perturbation on the camera orientation expressed in $\mathcal{F}_p$ of 18° about axis $\mathbf{u} = [0.61;\ -0.51;\ -0.61]$.

The set $\mathbf{d}_2$ includes the following perturbations:

- a perturbation of initial moving-platform pose of 0.13 m and 9.5° about axis $\mathbf{u} = [-0.52;\ 0.85;\ -0.04]$;
- a perturbation of camera pose in the moving-platform frame $\mathcal{F}_p$ of 0.05 m and 12.5° about axis $\mathbf{u} = [0.78;\ -0.51;\ -0.35]$;
- a perturbation of 0.005 m in a random direction for each cable exit point A_i and anchor point B_i.

The set $\mathbf{d}_3$ includes the following perturbations:

- a perturbation on the initial moving-platform pose of 0.075 m and 8° about axis $\mathbf{u} = [0.6;\ 0.78;\ -0.15]$;

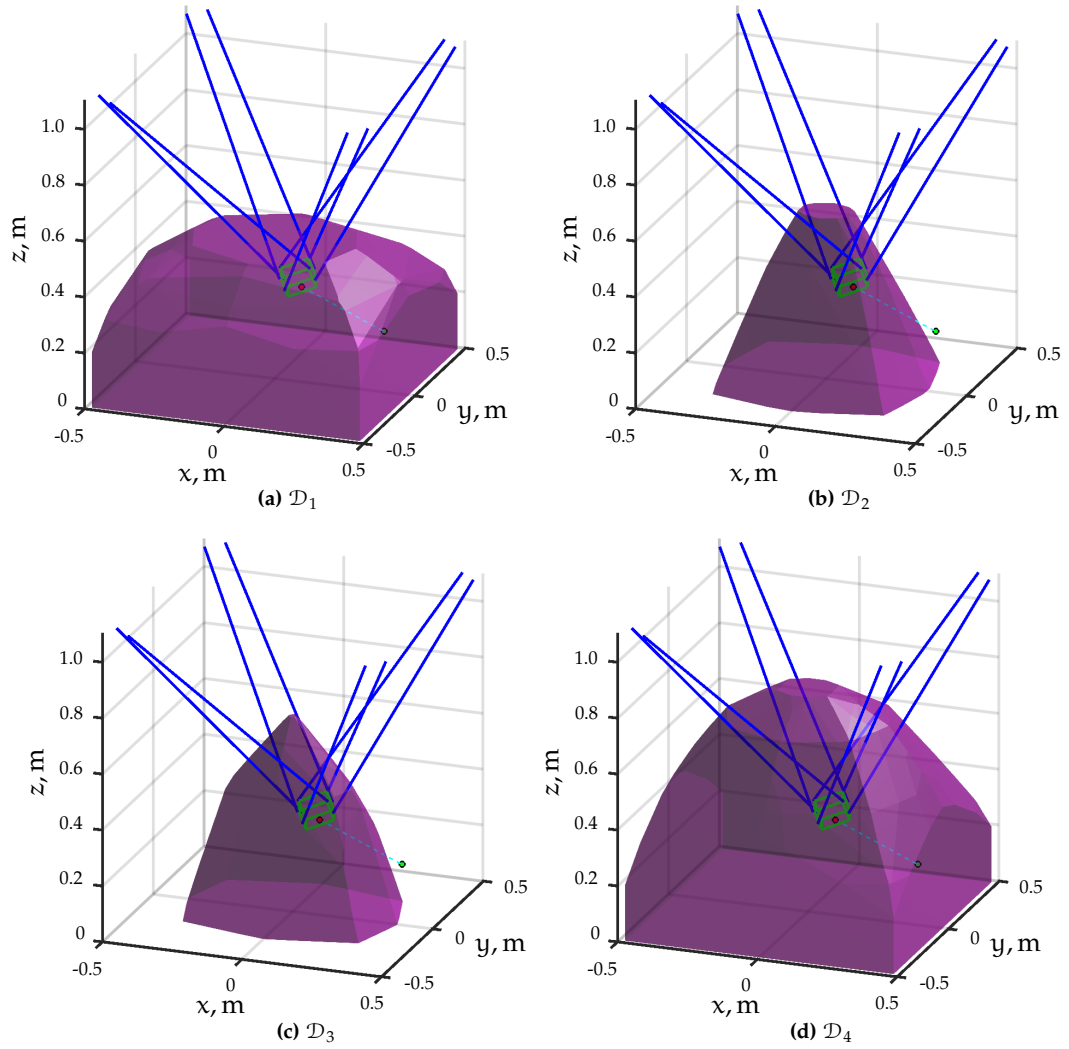


Figure 4.2: ACROBOT CSW for four perturbation bounds $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ and $\mathcal{D}_4$

- a perturbation on the camera pose in the moving-platform frame $\mathcal{F}_p$ of 0.05 m and 18° about axis $\mathbf{u} = [0.15;\ -0.63;\ 0.76]$;
- a perturbation for each cable exit point A_i and each cable anchor point B_i of 0.01 m along a randomly generated direction.

Finally, the set $\mathbf{d}_4$ includes the following perturbations:

- a perturbation on the initial moving-platform pose of 0.11 m and 6° about axis $\mathbf{u} = [-0.49;\ -0.5;\ 0.71]$;
- a perturbation on the camera orientation in the moving-platform frame $\mathcal{F}_p$ of 15° about axis $\mathbf{u} = [-0.47;\ -0.65;\ -0.60]$.

Clearly the set $\mathbf{d}_4$ belongs not only to $\mathcal{D}_4$, but also to $\mathcal{D}_1$. Note that the smaller workspaces correspond to $\mathcal{D}_2$ and $\mathcal{D}_3$, where perturbations on cable exit and anchor points are also taken into account.

The initial values are the following:

- ${}^{b}\mathbf{p}_{p_0} = \left[0.107\,\text{m};\ -0.026\,\text{m};\ 0.35\,\text{m};\ +20°;\ -20°;\ 0°\right]$
- ${}^{c}\mathbf{p}_{o_0} = \left[-0.022\,\text{m}; 0.136\,\text{m}; 0.449\,\text{m}; -157°; -18°; -176°\right]$
- $\mathbf{o} = \left[-0.043\,\text{m};\ 0.301\,\text{m}\right]$

and desired values are selected to be:

- ${}^{b}\mathbf{p}_{p^*} = \left[0.30\,\text{m};\ 0.25\,\text{m};\ 0.12\,\text{m};\ 0°;\ 0°;\ 0°\right]$
- ${}^{c}\mathbf{p}^*_o = \left[0\,\text{m};\ 0\,\text{m};\ 0.09\,\text{m};\ -180°;\ 0°;\ -180°\right]$
- $\mathbf{o}^* = \left[0\,\text{m};\ 0\,\text{m}\right]$

Note that the initial moving-platform pose is shown in Fig. 4.2, it is clearly within all of the workspaces. In two cases the desired moving-platform pose (shown with a green dot) is outside of the workspace. However, this does not mean that the system will become unstable. With the moving-platform very close to the desired state the error $\boldsymbol{e}$ will be small, the remaining motion in translation and rotation will be small as well, thus the system will be more robust to the perturbations. Indeed, the computed CSW is very conservative as it always assumes that (a) the distance to the goal is large; (b) each kind of perturbation range needs to be valid in all directions, while in an experiment only one direction is tested at a time.

For 2½D VS the adaptive gain λ, defined in (1.31), is used with the following coefficient values: $\lambda_0 = 2.0$, $\lambda_\infty = 0.4$ and $\dot{\lambda} = 30$. Furthermore, the behavior is improved by applying the gain $\sigma = 4.0$ to avoid velocity discontinuities as shown in (1.35).

For the controller with trajectory tracker, $\lambda = \lambda_0 = 2.0$ has been set, since the error is always small. Additionally, for the planner t_{full} is set to be equal to the execution time of the classic 2½D VS in order to ease comparability of the results. Finally, $\Delta t = 0.05\,\text{s}$.

4.3.2 Results

The very first experiment is a simulation in VREP and the results can be seen in Fig. 4.3. The trajectories of controllers without added perturbation are close to the ideal trajectories. Indeed, the error never surpasses 5 pixels or 7 millimeters, however it exists. The reason is that the VREP simulation includes slight cable elasticity, which is not taken into account in the control. Furthermore, the camera is not ideal, the frames it outputs are rather pixelated, which can lead to some small AprilTag localization errors.

When the perturbation set $\mathbf{d}_2$ is used, it is clear that the effect of the perturbations is more pronounced on the Cartesian trajectory of the camera, as can be seen in Figs. 4.3c and 4.3d. The maximum deviation of camera trajectory with 2½D VS is almost 60 mm, while the AprilTag center-point trajectory deviation does not reach 20 pixels. On the contrary, the controller with trajectory tracking successfully deals with the added perturbation and the deviation does not surpass 5 pixels and 10 mm.

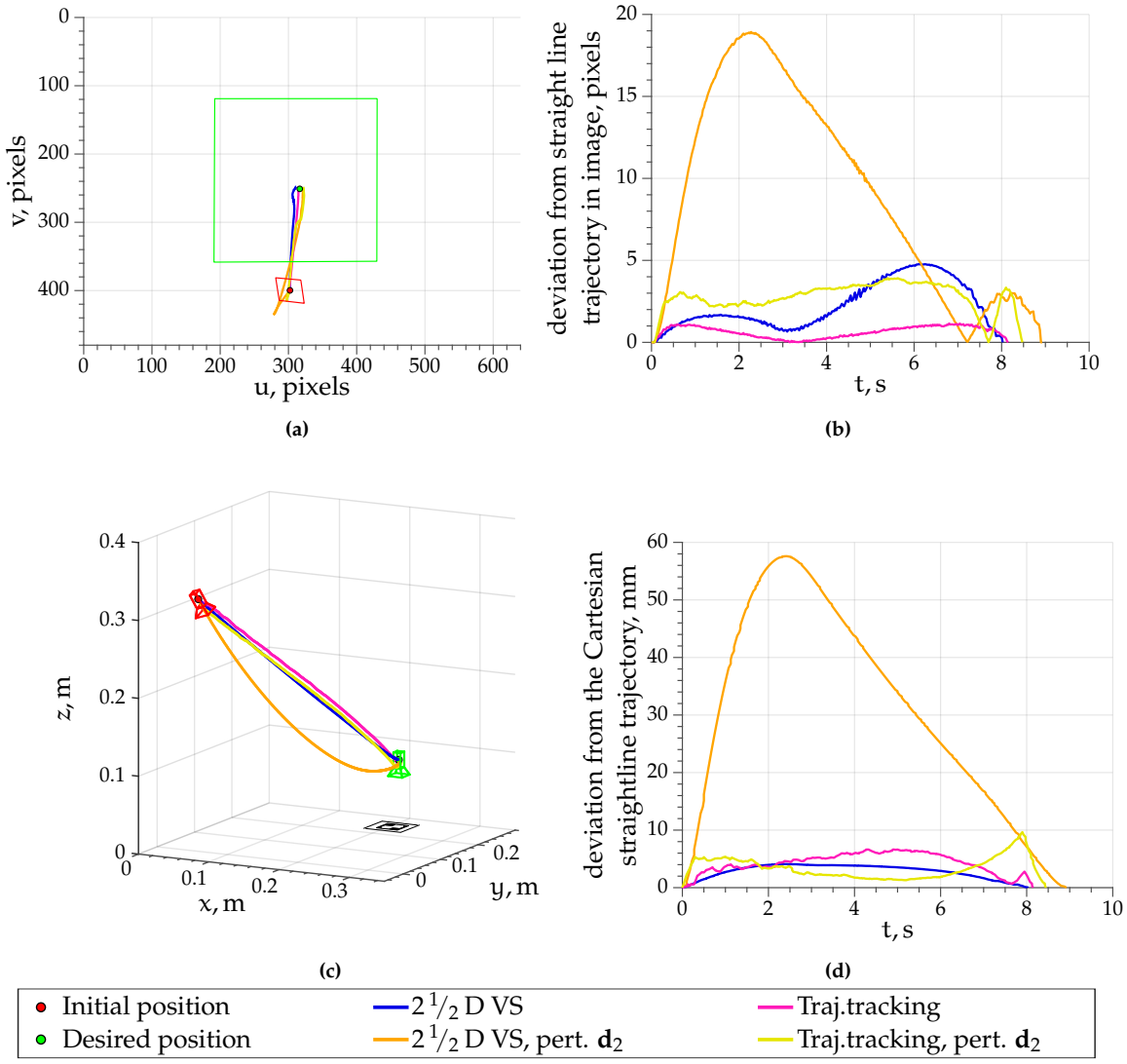


Figure 4.3: 2½D VS experiments in VREP: **(a)** the trajectory of AprilTag center-point in the image; **(b)** The pixel deviation from the ideal straight-line trajectory; **(c)** the trajectory of the camera in the frame $\mathcal{F}_b$; **(d)** the deviation from the ideal straight-line 3D trajectory.

The results from experiments on ACROBOT are shown in Figs. 4.4 and 4.5[6]. Note that the experiments with perturbations are divided in two pairs and are shown in two figures to avoid overly cluttered plots. Figures 4.4a and 4.5a show the trajectories of the AprilTag center-point in the image, while Figs. 4.4c and 4.5c show the 3D trajectories of the camera in the base frame $\mathcal{F}_b$. Additionally, the deviation from the straight-line trajectory in the image and in $\mathcal{F}_b$ is shown in Fig. 4.4b and Fig. 4.5b for perturbation sets $\mathbf{d_1}$ and $\mathbf{d_2}$; and Fig. 4.4d and Fig. 4.5d for perturbation sets $\mathbf{d_3}$ and $\mathbf{d_4}$, respectively.

Each controller, the classic 2½D VS and the one with trajectory tracking (named "Traj. tracking" in all of the figures) was tested without added perturbations and under the

[6]Please also see the accompanying video at `https://youtu.be/WsrWoH-Ping`

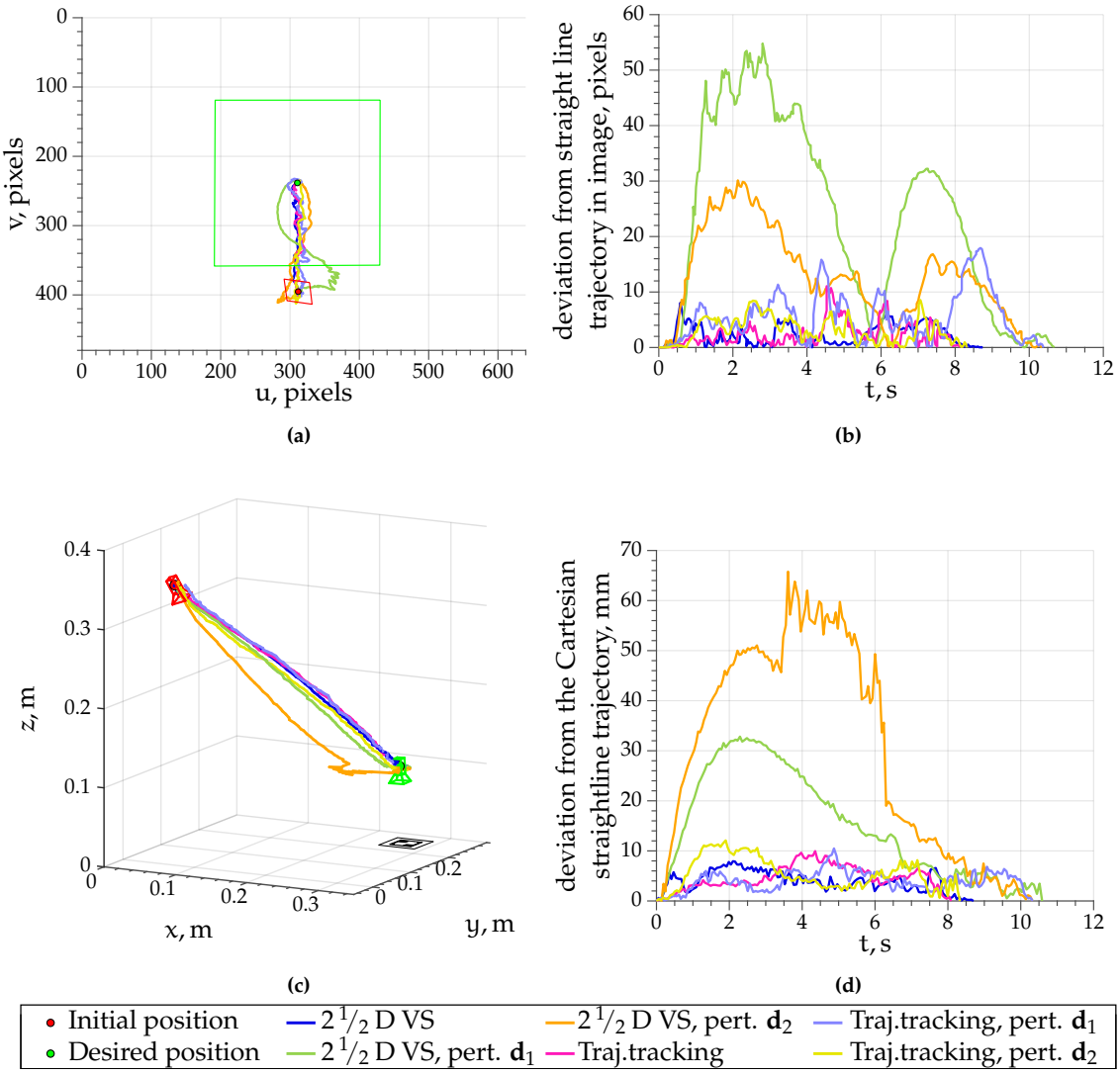


Figure 4.4: 2½D VS experiments on ACROBOT: **(a)** the trajectory of AprilTag center-point in the image; **(b)** The pixel deviation from the ideal straight-line trajectory; **(c)** the trajectory of the camera in the frame $\mathcal{F}_b$; **(d)** the deviation from the ideal straight-line 3D trajectory.

effect of each perturbation set $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$ and $\mathbf{d}_4$. Each experiment was repeated 15 times and the results are combined in a bar graph shown in Fig. 4.6.

While on a real CDPR the curves are not smooth even for the experiments without added perturbation, the behavior is in general similar to the simulation. Namely, under good conditions the trajectories are close to straight lines both in 3D and in the image. When no perturbation is added, the behavior of 2½D VS controller with and without trajectory tracking is similar. For both controllers the deviation does not surpass 0.01 m and 10 pixels. The superiority of trajectory tracking can be clearly seen when the system is perturbed. Each of the perturbation sets forces the classic 2½D VS to produce deviations from the ideal trajectories. $\mathbf{d}_2$ leads to highest deviation on the 3D trajectory (orange line in Fig. 4.4c), while $\mathbf{d}_1$ has the most pronounced effect on the trajectory in the image (light green line in Fig. 4.4a). The other two perturbation sets, shown in Fig. 4.5, lead to very

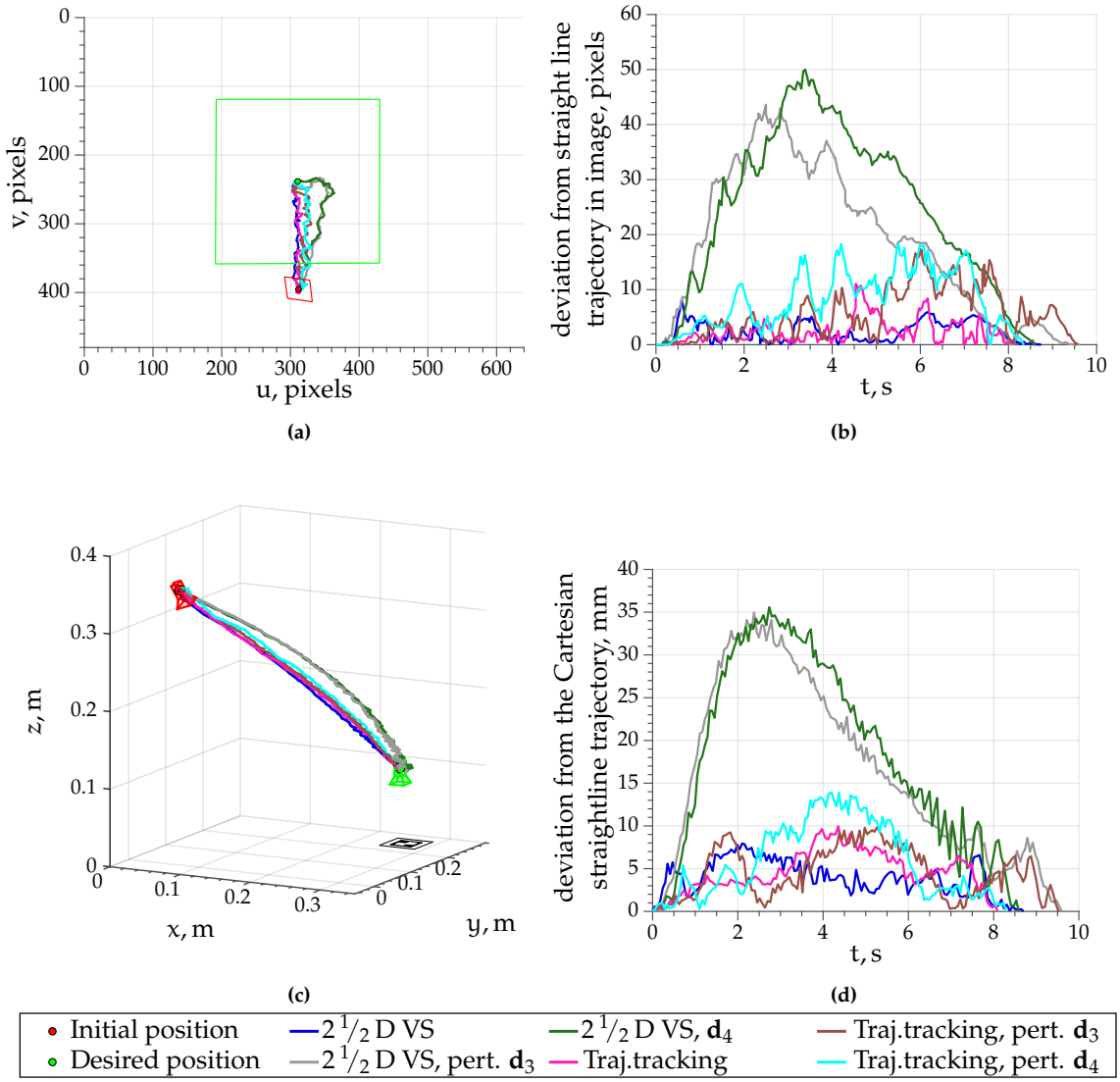


Figure 4.5: 2½D VS experiments on ACROBOT: **(a)** the trajectory of AprilTag center-point in the image; **(b)** The pixel deviation from the ideal straight-line trajectory; **(c)** the trajectory of the camera in the frame $\mathcal{F}_b$; **(d)** the deviation from the ideal straight-line 3D trajectory.

similar trajectory deviations even though the perturbation sets are clearly different. Indeed, depending on the direction of the perturbation of the same magnitude, the effect on the trajectories can be very different. And similarly, some very different sets of perturbations can be found that produce identical deviations.

On the contrary, the perturbation sets have a minimal effect on the trajectories produced by the controller with trajectory planning and tracking as depicted by the gray, yellow, brown and cyan lines in Figs. 4.4 and 4.5. Indeed, for the 3D trajectory the five lines corresponding to the trajectory tracking controller remain very near. The behavior is slightly worse in the image, where perturbation sets $\mathbf{d}_1$, $\mathbf{d}_3$ and $\mathbf{d}_4$ lead to about 18 pixel error (violet, brown and cyan lines, respectively). However, it is two to three times better

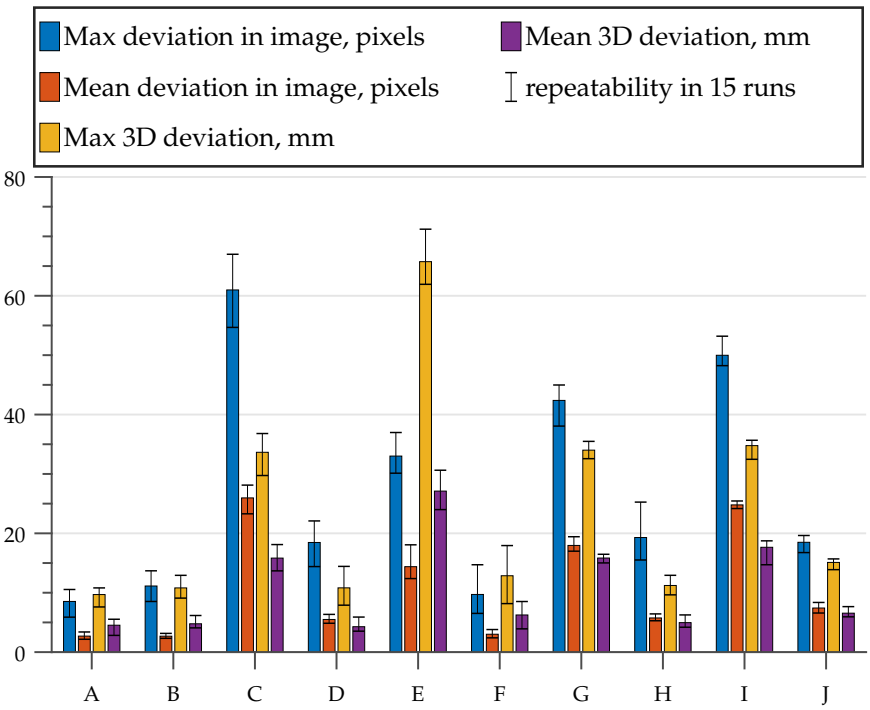


Figure 4.6: Bar graph showing the max and mean deviation from the ideal object center-point $\mathbf{c}$ trajectory in image and the ideal camera trajectory in $\mathcal{F}_b$ with and without voluntarily added perturbations. Classical 2½D VS A - without added perturbation; C - under the effect of perturbation set $\mathbf{d}_1$; E - under the effect of perturbation set $\mathbf{d}_2$; G - under the effect of perturbation set $\mathbf{d}_3$; I - under the effect of perturbation set $\mathbf{d}_4$. 2½D VS with trajectory planning and tracking: B - without added perturbation; D - under the effect of perturbation set $\mathbf{d}_1$; F - under the effect of perturbation set $\mathbf{d}_2$; H - under the effect of perturbation set $\mathbf{d}_3$; J - under the effect of perturbation set $\mathbf{d}_4$. Repeatability of results shown over 15 runs of each experiment.

than the deviation obtained with the classic 2½D VS under the same perturbations in Figs. 4.4b and 4.5b.

Figure 4.6 shows the maximum and the mean deviation from the ideal 2D and 3D trajectories for both controllers subjected to each of the perturbation sets. When there is no perturbation, the behavior of the controller without and with trajectory tracker is similar (groups A and B). No matter the perturbation set, the errors are at least two to three times smaller when the trajectory tracker is used: groups C and D for $\mathbf{d}_1$; groups E and F for $\mathbf{d}_2$; groups G and H for $\mathbf{d}_3$; groups I and J for $\mathbf{d}_4$. Furthermore, with trajectory tracking the 3D trajectory deviation (and the deviation of the trajectory in image for $\mathbf{d}_2$) remains similar to the experiment without perturbation shown as group B.

4.4 Case Study VIII: Trajectory Planning and Tracking on CAROCA

A short validation is done on CAROCA as well. As was shown in Section 3.4.4, even without additional perturbation, the produced AprilTag trajectory is rather curved most

likely due to not taking into account cable elasticity and pulley kinematics. Thus, it is of interest to see the CDPR behavior once trajectory planning and tracking is implemented.

4.4.1 Experimental Setup

To ease the comparison, the experimental setup from Section 3.4.4 is reused. For the trajectory tracker, the trajectory time t_{full} is set to 28 s with $\Delta t = 0.05$ s, and the constant gain λ is defined as $\lambda = \lambda_0 = 1.7$.

4.4.2 Results

The experimental results are shown in Fig. 4.7. The resulting trajectories generated by the controller with trajectory planner and tracker are shown in pink. Clearly the AprilTag center-point trajectory in Fig. 4.7a is a straight line. Indeed, the deviation is not perceptible as it remains less than 5 pixels throughout the trajectory, as can be seen in Fig. 4.7b. The improvement compared to classic 2½D VS is more than 5 times.

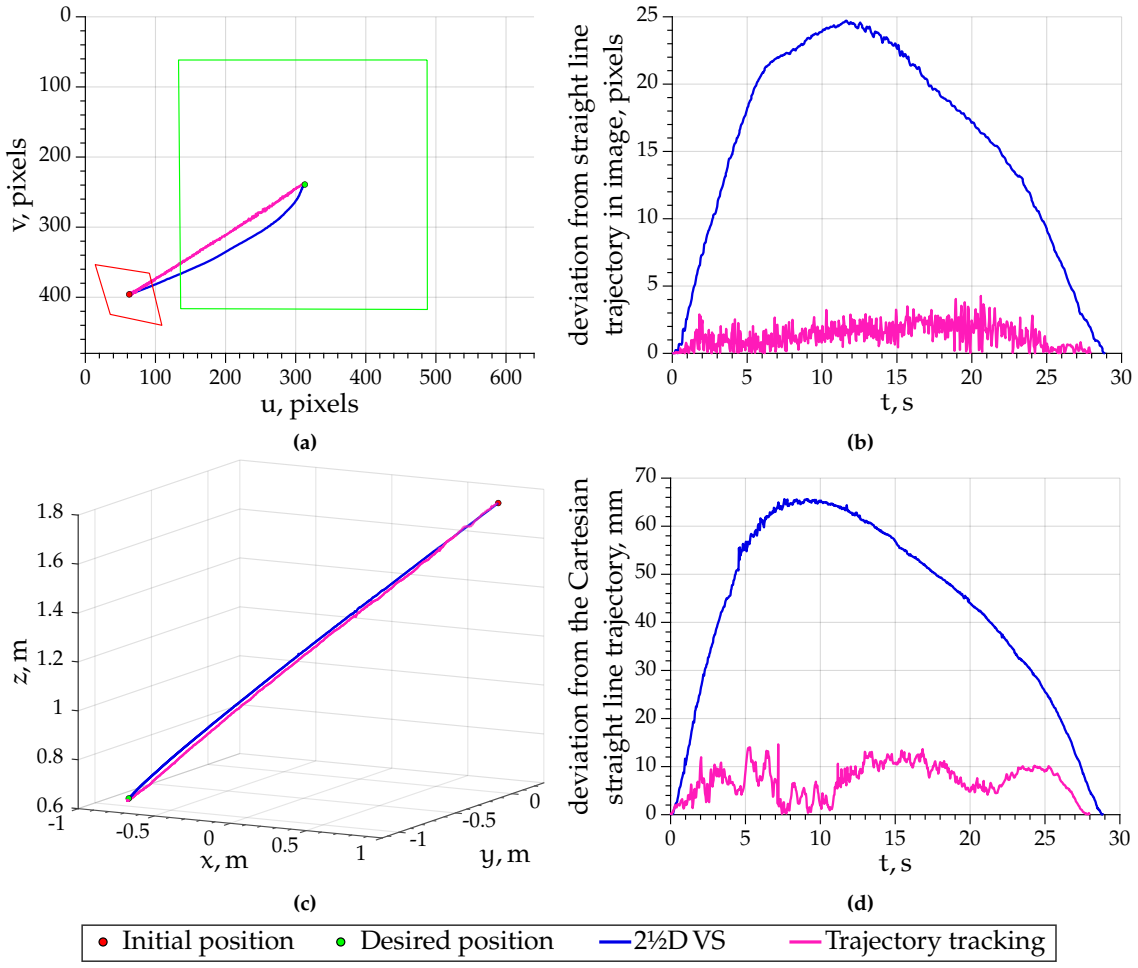


Figure 4.7: Trajectory planning and tracking experiments on CAROCA: **(a)** the trajectory of AprilTag center-point in the image; **(b)** The pixel deviation from the ideal straight-line trajectory; **(c)** the trajectory of the camera in the frame $\mathcal{F}_b$; **(d)** the deviation from the ideal straight-line 3D trajectory.

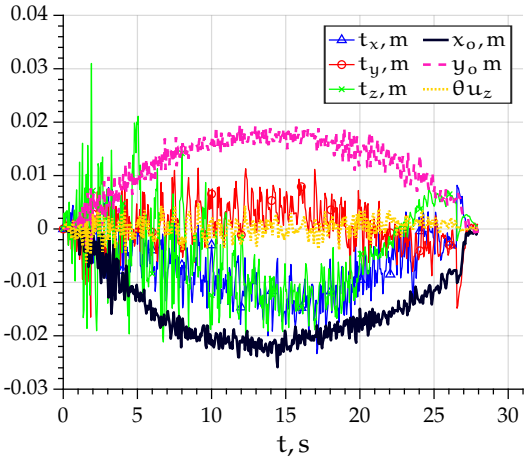


Figure 4.8: Error $\boldsymbol{e}$ plot for trajectory planning and tracking

The improvement of the camera trajectory in Fig. 4.7c is significantly less perceptible. This is because 2½D VS had less than 7 cm deviation for a 2.3 m distance. The controller with trajectory planner and tracker improves this behavior even more and now the deviation is no more than 1.5 cm, which is more than four times smaller.

It should be noted that the image deviation plot shown in Fig. 4.7b appears particularly noisy for the controller with trajectory planning and tracking. As can be seen in Fig. 4.8, multiple components of error $\boldsymbol{e}$ oscillate about value 0. It means that at two consecutive instants the computed cable velocities change significantly enough to produce a sort of moving-platform vibration. However, it is small enough to not influence the camera trajectory too greatly. As a reminder, it was measured by Creaform C-Track with accuracy of 0.02 mm (more details can be found in Appendix A.2). Thus, for CAROCA a smoothing filter should be applied to the computed error to obtain smoother control output.

4.5 Conclusions

This chapter proposed the use of trajectory planning and tracking with visual servoing. First, the proposed controller aims to increase the robustness of the system with respect to perturbations and errors in the robot model. Furthermore, it ensures that the produced trajectories are close to the ideal ones. It is implemented with 2½D VS and thus the straight-line motion of both the center-point of the AprilTag in the image and the camera in the base frame is ensured.

An extensive experimental validation was done. Several perturbation sets were selected for ACROBOT. It was found that depending on the perturbation direction, the behavior of the robot is different. Indeed, the amplitude and direction of trajectory deviation changed by changing the perturbation direction. Thus not only the range but also the direction of the perturbation matters.

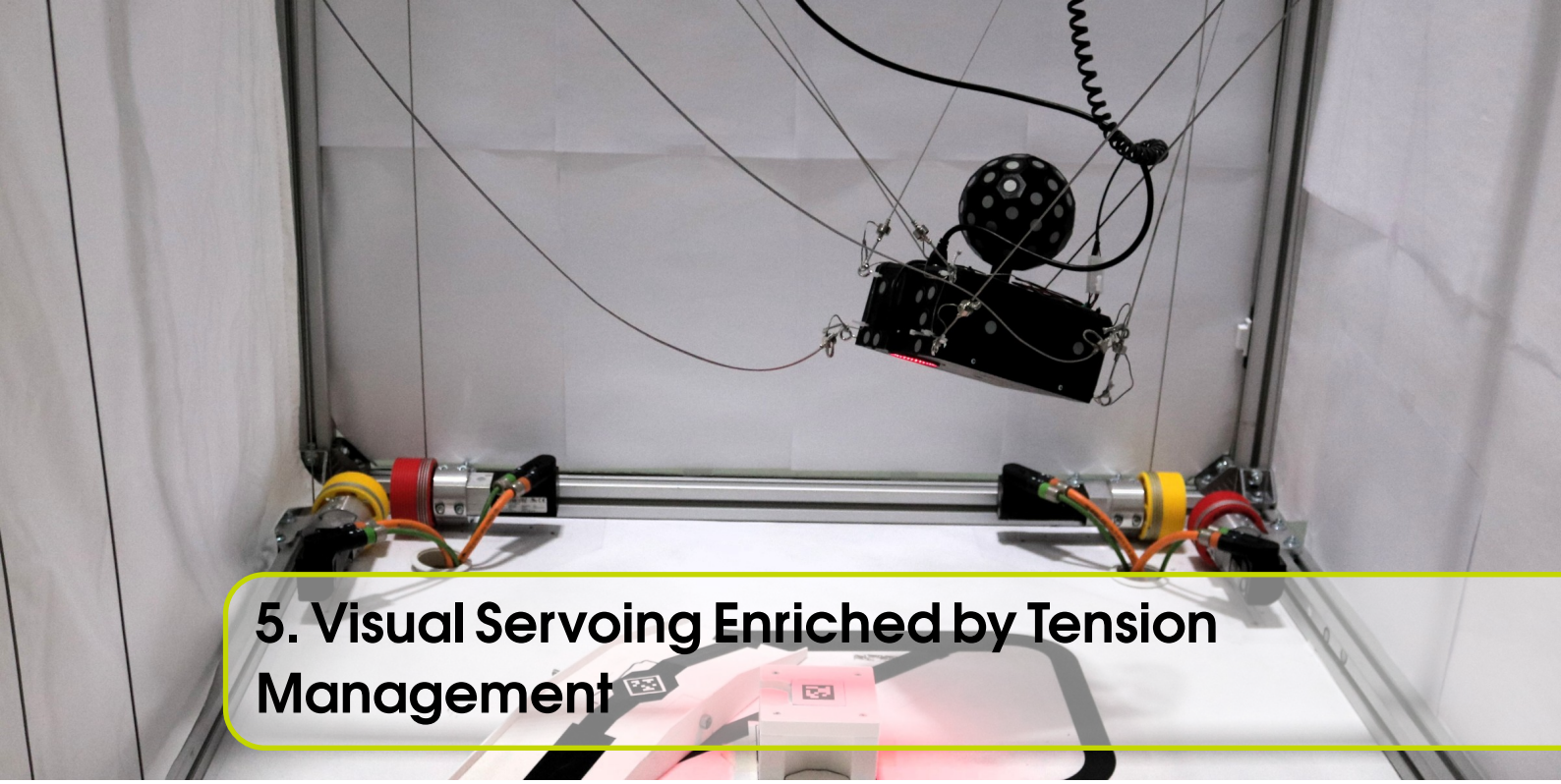
The improvement of robustness due to the use of trajectory planning and tracking was clearly shown in experimental validation on ACROBOT. While both systems, namely, without and with trajectory tracking, remain stable and achieve the goal, the trajectory produced by the former is clearly affected by perturbations.

CAROCA is a large CDPR with cables that tend to sag and pulleys of non-negligible diameter, thus in practice it differs from the simple CDPR model used in the control. This affects classic visual servo controllers and the produced trajectories are not ideal. However, using trajectory planning and tracking it is possible to obtain near-ideal output with unnoticeable deviation. Indeed, using 2½D VS with trajectory planning and tracking, the produced trajectory was nearly an ideal straight line for the camera and for the object center-point. More precisely, the deviation of the camera from the straight line over a 2.3 m distance is at most 1.5 cm and the mean is 0.71 cm with standard deviation of 0.34 cm.

In summary, as shown in our experiments, the addition of trajectory planning and tracking to the visual servoing controller allows us to keep the robot behavior close to the ideal one even in the presence of large perturbations. Such control of trajectories is especially important in an industrial environment or simply in a cluttered one. Indeed, as the trajectories are kept close to ideal, there is less chance of collisions with objects in the environment caused by perturbations.

However, it can occur that the perturbations are very high and even with trajectory planning and tracking, the deviations become significant. In such a case, when the system is very badly modeled/calibrated, it could occur that the current feature vector $\mathbf{s}(t)$ becomes very far from the desired one $\mathbf{s}^*(t)$. In this case it could be of interest to re-plan the trajectory. However, in practice the robot is able to withstand a perturbation of $\approx 0.2\,\mathrm{m}$ on the moving-platform position (combined with other perturbations), while its workspace is a $1\,\mathrm{m}^3$ cube. Therefore, it is highly unlikely that a re-planning step would be required for ACROBOT.

Note that any kind of trajectory can be planned. It does not necessarily have to be a straight line in Cartesian space or in the image. Thus, visual servoing with trajectory planning and tracking is an excellent approach to ensure not only goal accuracy, but also precise tracking of a predefined trajectory.



5. Visual Servoing Enriched by Tension Management

Contents

5.1 Introduction

In the previous chapters it was shown that visual servoing is a robust and accurate approach to control CDPRs. Using an onboard camera leads to high accuracy in the vicinity of the target while also providing robustness to different perturbations, including modeling errors. It was shown that the perturbations in the system do not affect the final accuracy, while they have an effect on the trajectory to the target. This issue was then successfully dealt with in Chapter 4. However, the implemented controllers do not deal with cable management. Indeed, it is assumed that as the CDPR executes the task, all of the cables are tensed and can thus pull on the moving-platform according to the control output.

In practice, cables can become slack during task execution or they can already be slightly slack at the beginning of the task. The slackness can occur for various reasons, such as: (i) CDPR modeling errors; (ii) coarse estimation of the initial moving-platform pose; (iii) reaching the border of SFW for the current orientation; (iv) cable creep (long-term extension); (v) coiling errors; (vi) having more cables than degrees of freedom of the moving-platform.

For a redundantly actuated CDPR, having r cables slack does not prevent the moving-platform from remaining in a given pose in a static configuration. Indeed, if cables are considered non-elastic then for a suspended CDPR at most six cables will be in tension [Mer17]. However, no matter the control approach, cable slackness leads to the CDPR being only partly responsive to the control output and thus not behaving as expected. Moreover, during a trajectory, cable slack will be transferred to different cables at the transition between two six-cable configurations [Mer17], the moving-platform becoming temporarily underactuated.

While some limited slackness does not make the system unstable, it acts as an additional perturbation. Indeed, if a cable is slack, applying a velocity to it will not produce the desired displacement of the moving-platform. A simplified example is shown in Fig. 5.1. Here, on the left a cable is in tension with initial length $l_i = 2\,\text{m}$ and after applying a velocity $\dot{l} = 0.12\,\text{m/s}$ for 1 s the final cable length is $l_f = 1.88\,\text{m}$ and the moving-platform has been moved by this cable. On the right the initial location of points A and B is the same as before, but the cable is now slack and its initial length is $l_i = 2.12\,\text{m}$. With the same cable velocity, the final cable length becomes $l_f = 2\,\text{m}$, but the moving-platform does not move. Indeed, the applied velocity only led to cable slack reduction and not yet to moving-platform motion.

With slack cables the actual displacement of the moving-platform differs from the estimated one. This can lead to accumulation of additional errors in moving-platform pose estimation for the methods based on control integration and on the CDPR model. This in turn leads to computation of cable velocities that can increase cable slackness. Thus, it is important to avoid cable slackness in order to improve the CDPR behavior and to keep its

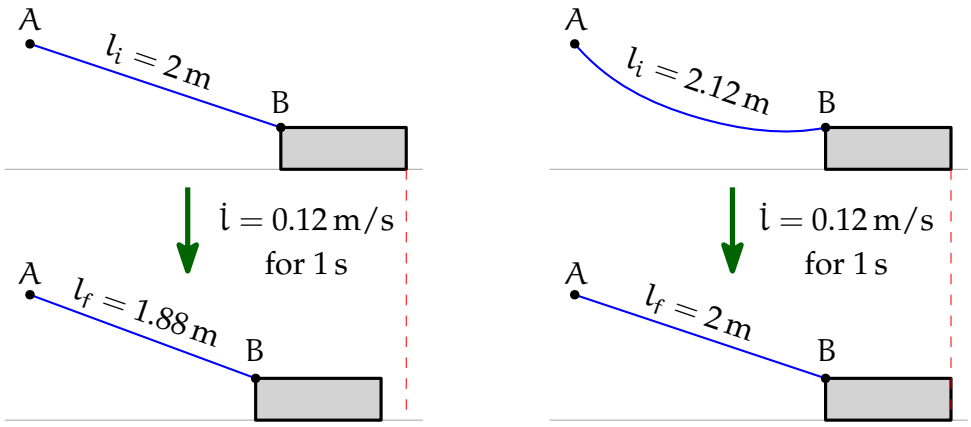


Figure 5.1: Effect of cable slackness on moving-platform displacement

moving-platform as stiff as possible. In this chapter different approaches to ensure cable tension management are proposed to enrich visual servoing controllers.

5.2 Tension Management Approaches

Cable tensions and moving-platform stiffness can be managed in multiple ways. First, cable tensions can be used in the CDPR model [Dal+12] [Dal+19]. This is unavoidable once a more complex cable model is to be considered, such as a sagging cable with non-negligible mass. In fact, the authors used not only tension sensors, but also four additional cameras to detect the cable angle that was also necessary for their chosen cable model. Note that using tensions in the control loop and not comparing them to a desired value does not ensure that there will be no slack in the system. While it is less likely thanks to the accurate cable model, cable slack can occur, especially if some cable characteristics are badly determined or due to perturbations in the system.

Usually, a Tension Distribution Algorithm (TDA) is used to determine the desired set of tensions for a desired moving-platform pose [CCL15] [Kra+14] [Ras+18]. The optimal tension distribution can be computed for all moving-platform poses in a path offline. These tensions are then used in control as output or they are transformed into motor torques for each iteration. If tension sensors are not available, the control is based on motor torques [CCL15]. Generally, extensive tuning is necessary to ensure good behavior and to avoid cable slack.

Ideally, if tension sensors are available, the control output is based on the difference between the TDA output and current tension measurements [Kra+14]. This allows to correct any cable slackness that might occur. This approach is not widely used because finding a good TDA solution online during the task execution can be problematic and special techniques are necessary [Bor+09] [Gou+15] [Pot14]. In [Kra+14] the main control is a position control and the tension measurements are compared to the output of a TDA and a cable length correction is produced. The final control output is a sum of

the inverse kinematics output and the correction. From their experiments, all cables of a fully-constrained CDPR are in tension and overall the cable tension values have tripled after correction. As a consequence, the stiffness of the manipulator is substantially increased.

Note that a position control scheme is developed in [Kra+14]. It is not directly usable for a velocity control. On the contrary, our goal is to keep using visual feedback in order to remain robust and accurate and to enrich this controller with tension management capability.

5.2.1 Visual Servoing with Tension-Based Correction of Velocity

Possibly the simplest approach is to keep the visual servoing controller as shown in Fig. 1.22 of Section 1.4.2 and to add a cable velocity correction that would deal with the cable slack. The proposed control scheme with a Tension Correction Algorithm (TCA) is shown in Fig. 5.2. This controller requires the addition of cable tension sensors.

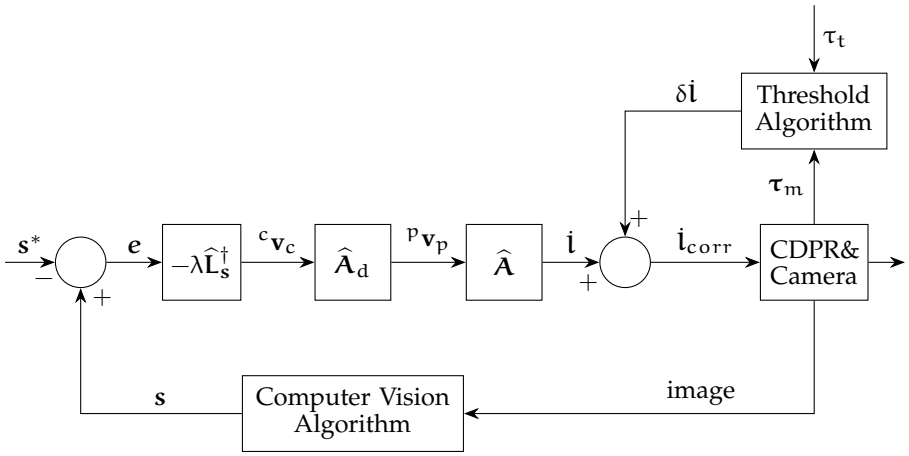


Figure 5.2: Visual servoing control of a CDPR with a TCA

Here, at every iteration cable tensions are measured and recorded in vector $\boldsymbol{\tau}_m$. Each component of this vector is then compared to a threshold tension τ_t and the ith component $\delta\dot{l}_i$ of a cable velocity correction vector $\delta\dot{\mathbf{l}}$ is computed as:

$$\delta\dot{l}_i = \begin{cases} -k_c(\tau_t - \tau_{m_i}) \text{ if } \tau_{m_i} < \tau_t, \;\; i = 1, \ldots, m \\ 0 \text{ otherwise} \end{cases} \tag{5.1}$$

where τ_{m_i} is the measured tension of the ith cable, and k_c is a positive gain that needs to be tuned. The tension τ_t is the threshold value under which the cable is considered to be slack.

The resulting behavior depends on the sign of the ith cable velocity $\dot{l}_i$ computed by the controller. Note that negative velocity leads to the cable becoming shorter, and positive velocity leads to the cable becoming longer. Furthermore, according to (5.1) $\delta\dot{l}_i$

is not positive. Thus applying $\delta\dot{l}_i$ to a negative $\dot{l}_i$ results in cable length reducing faster. Applying $\delta\dot{l}_i$ to a positive $\dot{l}_i$ results in cable length increasing slower.

It should be noted that the correction speed is a function of the gain k_c. The greater the gain, the faster the correction. However, setting k_c too high may perturb the main controller. Thus, k_c should be tuned based on the frequency of the TCA loop. Furthermore, k_c is used to change the order of magnitude between the tension difference and cable velocities. For example, if $(\tau_t - \tau_{m_i}) \approx 1\,\mathrm{N}$ and $\dot{l}_i \approx 0.05 = 5 \times 10^{-2}\,\mathrm{m/s}$, then k_c will be defined as $k_c = 10^{-2}\,\mathrm{m/Ns}$.

The threshold τ_t is a tension that is feasible for every cable no matter the moving-platform pose. Such a tension can be found by tracing a workspace, e.g. the Static Feasible Workspace (SFW), presented in Section 1.3.3, and choosing the lower tension bound τ_{lb}. Indeed, τ_{lb} is feasible for every cable and for all moving-platform poses within SFW, thus it is a good pick for τ_t.

The proposed tension correction algorithm is greatly simplified when compared to [Kra+14]. Indeed, instead of using a TDA and a complex computation of the final tension correction, we simply compare the current tension measurement to a threshold. In this case, the need for the knowledge of the moving-platform pose is avoided since the tension threshold is constant. It is very convenient, because we only have a coarse moving-platform pose estimation. Furthermore, using a coarse moving-platform pose estimation with a TDA could be impossible. More precisely, the tension set provided by the TDA could be unattainable with the current moving-platform pose in case of estimation errors. This would lead to perturbing the visual servo controller and possibly even making it unstable, thus failing the task. In addition, the simpler the calculations, the faster the controller response to each incoming image.

In case TCA is not efficient enough due to its simplicity, there are several ways to extend it as shown in the following sections. Note that these approaches will require a better knowledge of at least the initial moving-platform pose.

Gain as a Function of Cable Lengths

To increase the slack correction rate, it could be possible to set the gain as a function of cable lengths. More precisely, the gain should depend on the cable length error.

This approach would require a good moving-platform pose estimation or ideally its measurement. Given a moving-platform pose, it is possible to compute the expected cable length vector $\mathbf{l}_e$. It can then be compared to the actual cable length vector $\mathbf{l}_a$. The latter is retrieved from the motor positions $\mathbf{q}_m$ and the initial cable length vector $\mathbf{l}_0$:

$$\mathbf{l}_a = \mathbf{l}_0 + r_w \mathbf{q}_m \tag{5.2}$$

Note that here it is assumed that at $t = 0\,\mathrm{s}$ when $\mathbf{l}_a = \mathbf{l}_0$, the motor positions $\mathbf{q}_m$ are reinitialized to zero.

Then an adaptive gain $\mathbf{k}_a$ can be expressed as:

$$\mathbf{k}_a = k_c \delta \mathbf{l} \tag{5.3}$$

where $\delta \mathbf{l} = \mathbf{l}_a - \mathbf{l}_e$ and k_c is the previously defined constant gain.

The obtained gain vector $\mathbf{k}_a$ is used to compute the velocity correction $\delta \dot{\mathbf{l}}$:

$$\delta \dot{l}_i = \begin{cases} -k_{a_i}(\tau_t - \tau_{m_i}) \text{ if } \tau_{m_i} < \tau_t, \;\; i = 1, \ldots, m \\ 0 \text{ otherwise} \end{cases} \tag{5.4}$$

Consequently, cable velocity correction only occurs when cable slack is detected by the tension sensors. However, with the adaptive gain, the correction rate increases with the increased slack. Thus, the correction of 1 cm and 8 cm slack should happen in the same amount of time. Rapidly correcting cable slack allows us to reduce the probability of underactuation during transfer of slack between different cables. The drawback remains that this approach requires a rather good knowledge of moving-platform pose in order to avoid creating incorrect corrections and thus the use of another sensor at least at the very beginning of the task.

Threshold τ_t as a Function of Cable Lengths

Setting τ_t as a function of cable length could increase the stiffness of the moving-platform. In fact, this can be considered a middle step between TCA and TDA.

It could be assumed that the shorter the cable, the higher the tension to be applied by that cable. This assumption is based on the fact that the shorter the cable C_i, the closer the moving-platform to the exit point A_i and thus the larger the part of the moving-platform mass that is supported by this cable.

First of all, the higher limit of the threshold needs to be found along with the corresponding cable length. Note that with the same cable length but different cable angle the tension measurement will be different. Indeed, a vertical and a horizontal cable of same length will not have the same tension. Thus it appears that τ_t should be a function of cable length and cable angle. The latter however can be difficult to obtain without additional sensors. Indeed, it can be estimated from the moving-platform pose, but that requires a good knowledge of the pose.

Another concern is the relation between cable length and cable tension. It could be defined as simply linear or exponential. The risk with such an approach is that if the threshold is set too high, the correction algorithm will attempt to correct a false-slack cable and by doing so will pull on the moving-platform. This in turn can make another cable register as slack and the TCA would start correcting it too. This behavior would make the whole system become unstable. The tuning of such a system would be very complex.

To conclude, such an approach is indeed possible, but would require a careful tuning and use of additional sensors. It stands to question whether the improvement, which may come, would be worth the increased complexity. It should also be noted that the feasible polygon is usually very small for suspended CDPRs. Therefore, the size of possible cable tension space is small. Accordingly, selecting a set of cable tensions at the barycenter of the feasible polygon or at its boundary (the current solution) will not change the behavior of the CDPR that much. Of course, this is not true for fully-constrained CDPRs and there a constant threshold will not be optimal.

Obtaining τ_t Through TDA

It should be noted that it is not possible to use directly a TDA instead of a TCA, because the final control is in velocity. Instead TDA could be used to provide a tension vector $\boldsymbol{\tau}_t$ for a given moving-platform pose. An updated control scheme is shown in Fig. 5.3.

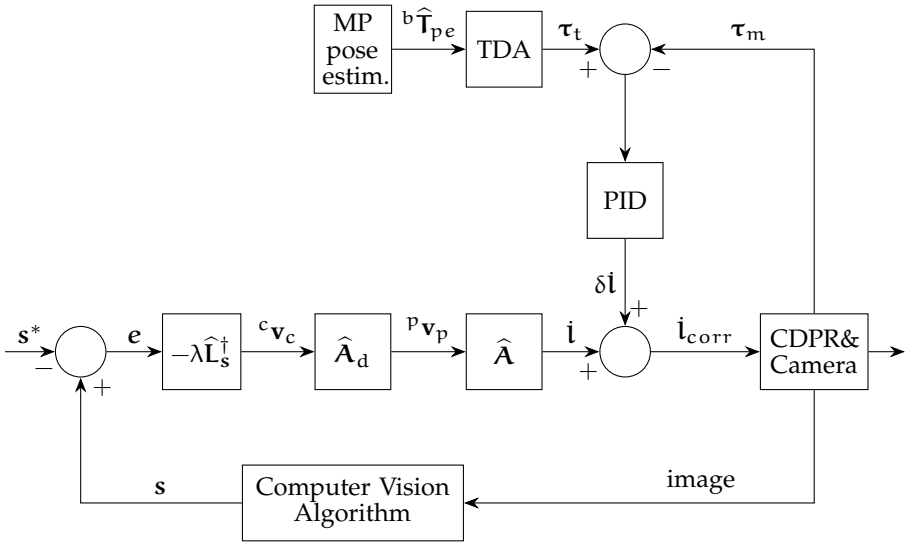


Figure 5.3: Visual servoing control of a CDPR with a TDA

The TDA block requires the next expected moving-platform pose as an input. Indeed, we want to compute the tension distribution corresponding to the pose where the moving-platform would arrive after Δt seconds. Thus, it is necessary to use control-based moving-platform pose estimation in the block *MP pose estim.* as shown in Section 2.2.1. The expected moving-platform pose is denoted as $^b\widehat{\mathbf{T}}_{pe}$. The control scheme contains a *PID* block, but initially just a proportional controller could be used. Indeed, in such a case the proportional gain would be similar to k_c, which was defined for TCA.

Evidently, this scheme is more complex. Furthermore, we suspect that it will be less robust to moving-platform pose estimation errors. This is because the moving-platform pose is required by the TDA to be able to compute a corresponding tension set $\boldsymbol{\tau}_t$. To be able to fully take advantage of the TDA, a moving-platform pose measurement should be added to the control. In such a case, one could assume that the moving-platform pose

is well known and attempting to attain the computed tension set $\boldsymbol{\tau}_t$ will not perturb the main controller.

5.2.2 Visual Servoing with Cable Length-Based Velocity Correction

A similar cable length-based approach can be created. In this case the cable velocity correction vector $\delta\dot{\mathbf{l}}$ is computed based on cable lengths. The control scheme is shown in Fig. 5.4.

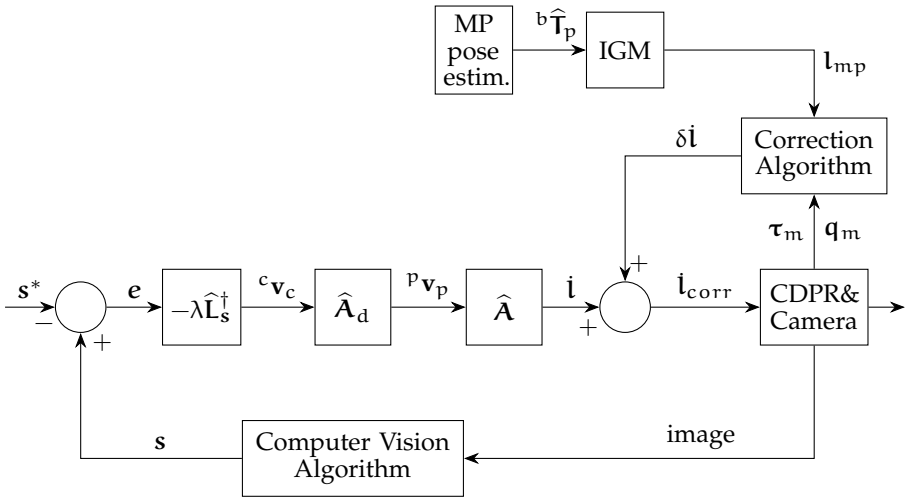


Figure 5.4: Visual servoing control of a CDPR with a TCA

More precisely, the current tension measurements $\boldsymbol{\tau}_m$ and motor positions $\mathbf{q}_m$ are retrieved from the CDPR. Current cable lengths $\mathbf{l}_a$ are computed with (5.2). A second set of cable lengths $\mathbf{l}_{mp}$ is computed via the IGM for the estimated moving-platform pose in matrix form $^{b}\widehat{\mathbf{T}}_p$. Note that here we estimate the current moving-platform pose, unlike Fig. 5.3. Using the tension measurements $\boldsymbol{\tau}_m$ we find the two cables $\mathcal{C}_{b1}$ and $\mathcal{C}_{b2}$ with the lowest tension and the velocity correction is computed as:

$$\delta\dot{l}_i = \begin{cases} (l_{mpi} + l_t - l_{ai})/(k_l \Delta t) & \text{if} \quad l_{mpi} + l_t < l_{ai} \ \text{ and } \ (\mathcal{C}_i = \mathcal{C}_{b1} \ \text{ or } \ \mathcal{C}_i = \mathcal{C}_{b2}) \\ 0 \quad \text{otherwise} \end{cases} \tag{5.5}$$

where l_{mpi} is the ith element of $\mathbf{l}_{mp}$; l_{ai} is the ith element of $\mathbf{l}_a$; $k_l \geqslant 1$ is a gain that needs to be tuned; l_t is a non-negative threshold cable length that can be set to 0.

Thus, the correction is only computed for the two cables with the lowest tension measurement and only if the actual length l_{ai} is longer than the correct length for the current moving-platform pose l_{mpi} summed with a safety threshold l_t. In a perfectly known system l_t would be set to zero, however in our case it should be a positive value with the initial guess being $l_t = 0.01$ m. The gain k_l is used to adjust the correction rate. Ideally it could be set to $k_l = 1$, however it would be safer to choose a larger value.

The risk with such a scheme and constant gain values is to never remove slack completely. An adaptive gain could be used instead, as described in the improvement propositions of Section 5.2.1.

5.2.3 Visual Servoing with Torque Control

Instead of controlling the robot in velocity, it can be done also with motor torques. In this case the camera velocity computed by the visual servo controller needs to be transformed into the expected moving-platform pose ${}^{b}\widehat{\mathbf{T}}_{pe}$ to be used in the TDA block. The proposed control scheme is shown in Fig. 5.5.

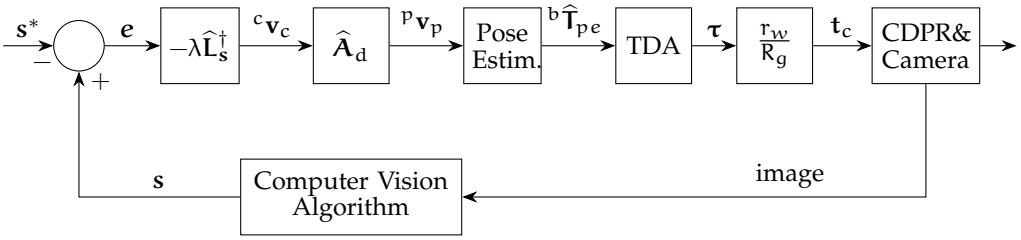


Figure 5.5: Vision-based torque control

The TDA, presented in Section 1.3.4, computes the desired set of tensions $\boldsymbol{\tau}$ that corresponds to the given moving-platform pose. The tensions $\boldsymbol{\tau}$ need to be transformed into motor torques, which is done simply by defining $\mathbf{t}_c = \frac{r_w}{R_g}\boldsymbol{\tau}$, where r_w is the winch radius and R_g is the gearbox reduction rate.

However, this controller cannot be used directly, because the motor torques computed from the cable tensions are not sufficient to surpass motor friction. Indeed, the final torque command $\mathbf{t}_m$ given to the robot must include a second component covering the motor friction, as shown in Fig. 5.6:

$$\mathbf{t}_m = \mathbf{t}_c + \mathbf{t}_f \tag{5.6}$$

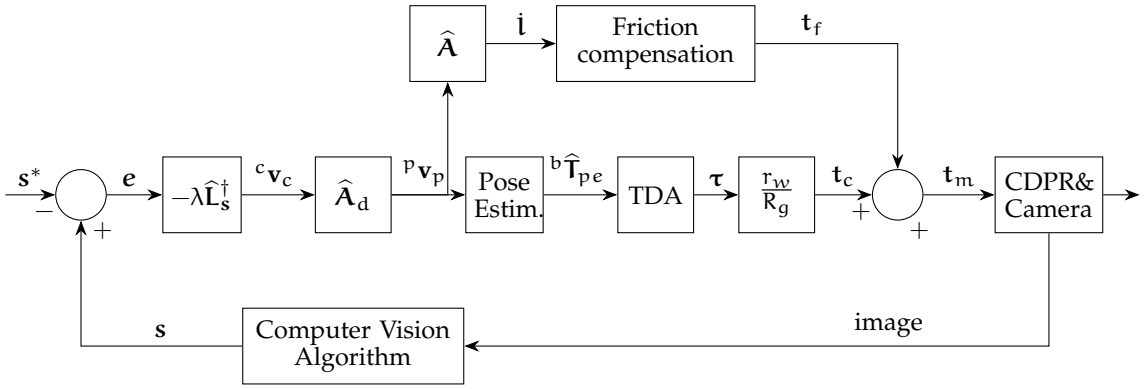


Figure 5.6: Vision-based torque control with motor friction

where

$$\mathbf{t}_f = F_c + (F_s - F_c)e^{(-\dot{\mathbf{q}}_m/\omega_s)} + F_v\dot{\mathbf{q}}_m \tag{5.7}$$

with F_c being the Coulomb friction coefficient, F_s being the static friction coefficient, F_v being the viscous friction coefficient, ω_s being the Stribeck velocity, and $\dot{\mathbf{q}}_m$ is the motor velocity vector, which can be obtained from cable velocities $\dot{\mathbf{l}}$ as:

$$\dot{\mathbf{q}}_m = \frac{1}{r_w}\dot{\mathbf{l}} \tag{5.8}$$

The determination of friction coefficients F_c, F_s and F_v must be done experimentally for each motor. The torque measurements for each motor at different velocities need to be recorded. Then, by tuning the friction coefficients, the curve is fitted to the recorded torque measurements.

One should also note that motor torques that are retrieved from motor encoders are often very noisy. Due to this it would be preferable to use, e.g., tension sensors and a PID to control the robot, as shown in Fig. 5.7. Here, the measured tensions $\boldsymbol{\tau}_{meas}$ are compared to the tension vector $\boldsymbol{\tau}$ produced by the TDA. The difference is then used in the PID controller. The latter outputs the motor torques $\mathbf{t}_m$.

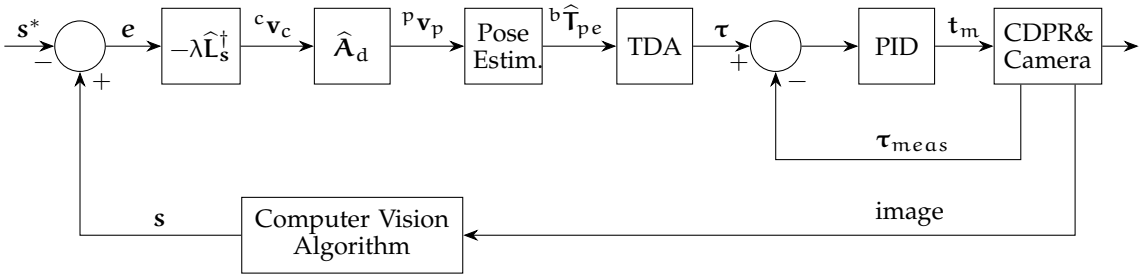


Figure 5.7: Vision-based tension control with a PID

Finally, such a controller suffers from the fact that the moving-platform pose estimation is open-loop. Indeed, the currently available sensors, namely the onboard camera and tension sensors, do not allow us to measure the moving-platform pose. Instead, it needs to be estimated. Once the estimation becomes too coarse, the robot will not be able to arrive at the desired location. This is because the computed tension set $\boldsymbol{\tau}$ corresponds to the estimated moving-platform pose. In fact, it could happen that this tension set is not within the feasible polygon of the actual moving-platform pose. Thus, ideally this controller would need to be enriched by a measurement system that would retrieve the moving-platform pose. For example, a second camera, used in the eye-to-hand configuration, can be used to localize the moving-platform in the Cartesian space. The measured moving-platform pose could then be given to the *Pose Estim.* block to compute the desired moving-platform pose at this step, as can be seen in Fig. 5.8.

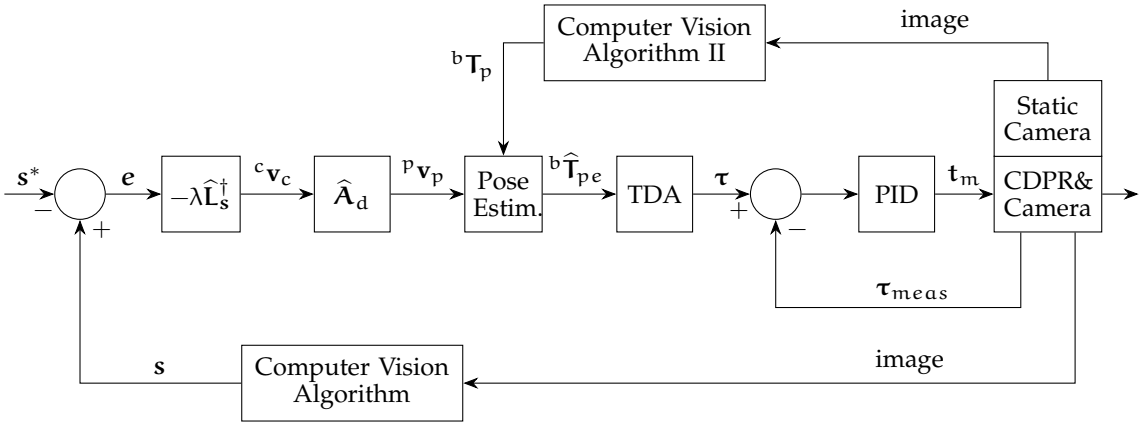


Figure 5.8: Vision-based tension control with a PID and an eye-to-hand camera

5.2.4 Vision-Based Position and Torque Control

In the previous section the focus was on keeping the advantages of visual servoing and transforming the control signal into torque control, while passing by the TDA. Here, we propose abstracting from the classic visual servoing controller even more. There are two options, either position or torque control can be used. As can be seen in Figs. 5.9 and 5.10, the approach is basically the same.

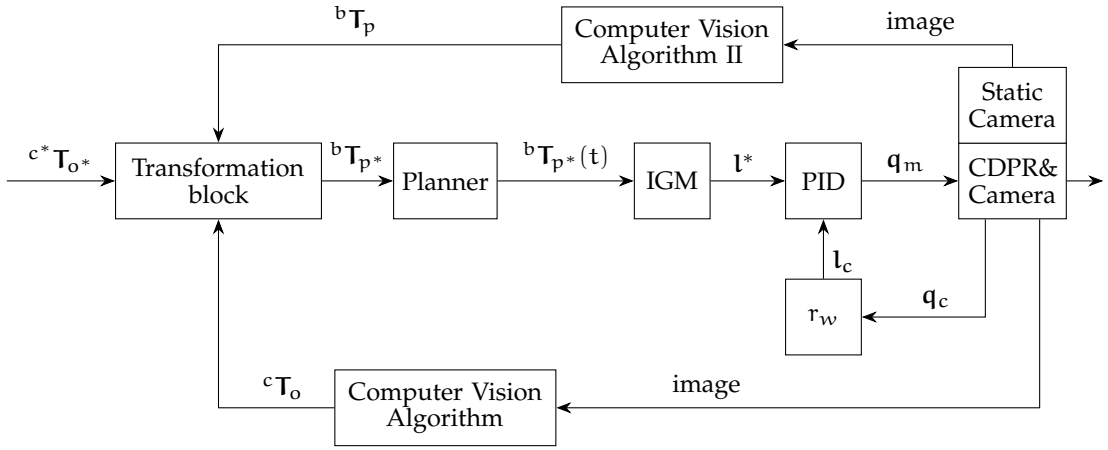


Figure 5.9: Position control based on object pose measurements

It is assumed that the object pose in the camera frame can be retrieved from the Computer Vision Algorithm. This pose is expressed as a transformation matrix $^{c}\mathbf{T}_{o}$. Similarly, the desired object pose is defined as the desired transformation matrix $^{c^{*}}\mathbf{T}_{o^{*}}$. The current moving-platform pose is retrieved from a static camera and the transformation matrix $^{b}\mathbf{T}_{p}$ is defined. Then the desired moving-platform pose is found as:

$$^{b}\mathbf{T}_{p^{*}} = {}^{b}\mathbf{T}_{p}\,{}^{p}\mathbf{T}_{c}\,{}^{c}\mathbf{T}_{o}\,{}^{c^{*}}\mathbf{T}_{o^{*}}^{-1}\,{}^{c}\mathbf{T}_{p} \tag{5.9}$$

Since it is highly likely that $^{b}\mathbf{T}_{p^{*}}$ is far from $^{b}\mathbf{T}_{p}$, then giving the desired moving-platform pose directly to IGM or TDA can lead to very high cable velocities

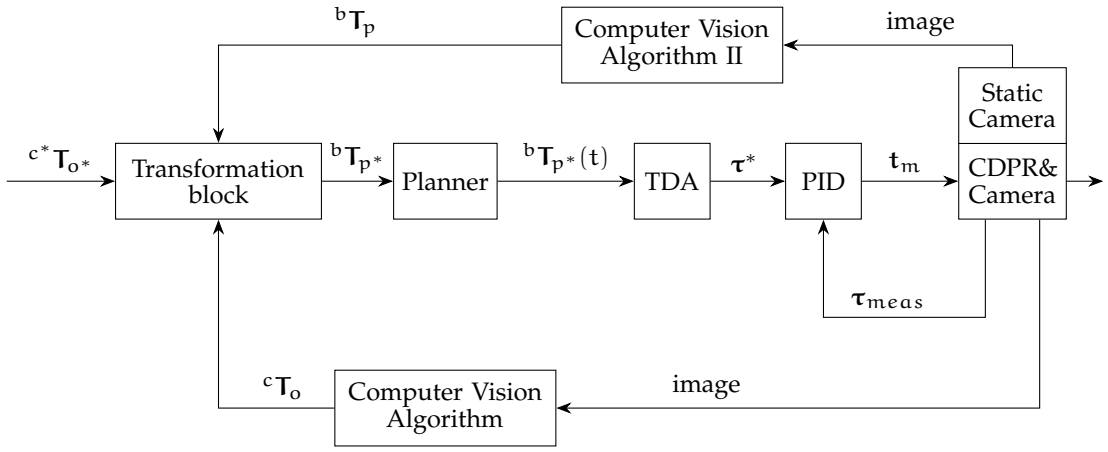


Figure 5.10: Torque control based on object pose measurements

and tensions. Instead a planner needs to be used to find a $^{b}\mathbf{T}_{p^*}(t)$ that is not too far from $^{b}\mathbf{T}_{p}$ and would ensure smooth transition from the initial state to the desired one.

In case of the position controller, $^{b}\mathbf{T}_{p^*}(t)$ is given to the IGM, which outputs the desired cable length vector $\mathbf{l}^*$. The latter is compared to the current cable length vector $\mathbf{l}_c$ in the PID block. Finally, PID outputs the motor positions.

In case of the torque controller, $^{b}\mathbf{T}_{p^*}(t)$ is given to TDA, which outputs the desired tension vector $\boldsymbol{\tau}^*$. The latter is compared to the measured cable tensions $\boldsymbol{\tau}_{meas}$ in the PID block and the CDPR receives the resulting torque vector $\mathbf{t}_m$.

In both cases it is assumed that the inner loop with the PID has a significantly higher frequency than the outer loop with cameras. Indeed, the onboard camera is used to update the current knowledge of the object pose in the camera frame. This in combination with the current moving-platform pose allows us to correct the desired moving-platform pose.

It should be noted that with such a controller it is highly likely that the target object leaves the camera field of view. Indeed, here the control output is generated by a PID and not by visual servoing. For this reason the object can leave the image frame and consequently, the straight-line motion of the object in the image is not ensured.

5.3 Implementation of TCA on two CDPRs

In the following sections the TCA is validated on both ACROBOT and CAROCA. Considering the very good results obtained, the more complex methods have not been implemented.

5.3.1 Case Study IX: VS with TCA on ACROBOT

Experimental Setup

The small CDPR prototype ACROBOT is used to validate experimentally the effect of cable slackness on the system behavior. The large moving-platform is used, because

it houses the tension sensor hardware. Tension sensors are mounted on the cables close to their anchor points B_i. They are calibrated for a range from $-25\,\mathrm{N}$ to $75\,\mathrm{N}$, with an accuracy of $0.24\,\mathrm{N}$ and repeatability of $0.01\,\mathrm{N}$. Of course a cable cannot compress, however it was not possible to calibrate the sensors only for tension.

The main loop frequency is $25\,\mathrm{Hz}$. The TCA loop can be as fast as $512\,\mathrm{Hz}$, but for the following experiments it was kept at main loop frequency. The tuning of gain λ is described in Section 4.3. After tracing SFW as shown in Section 1.3.3, it was found that the optimal lower bound of cable tension is $\tau_{lb} = 1\,\mathrm{N}$, because the workspace remains large and all cables remain in tension. Thus, we set $\tau_t = \tau_{lb} = 1\,\mathrm{N}$. Finally, TCA gain k_c is tuned to be $0.04\,\mathrm{m/Ns}$.

Experiments with Cable Slackness at the Initial Moving-Platform Pose

Every experiment starts at a predefined initial moving-platform pose corresponding to the initial feature vector $\mathbf{s}$. Similarly, the desired moving-platform pose, corresponding to the desired feature vector $\mathbf{s}^*$, is the same for all experiments.

The initial values are the following:

- ${}^b\mathbf{p}_{p0} = \left[0.202\,\mathrm{m};\ 0.118\,\mathrm{m};\ 0.268\,\mathrm{m};\ -18°;\ 10°;\ 4°\right]$
- ${}^c\mathbf{p}_{o0} = \left[0.1\,\mathrm{m};\ -0.05\,\mathrm{m};\ 0.26\,\mathrm{m};\ 165°;\ 10°;\ 179°\right]$
- $\mathbf{o}_0 = \left[0.38\,\mathrm{m};\ -0.19\,\mathrm{m}\right]$

and desired values are selected to be:

- ${}^b\mathbf{p}_{p^*} = \left[-0.11\,\mathrm{m};\ -0.20\,\mathrm{m};\ 0.366\,\mathrm{m};\ 13°;\ -20°;\ 33°\right]$
- ${}^c\mathbf{p}_o^* = \left[-0.14\,\mathrm{m};\ 0.115\,\mathrm{m};\ 0.35\,\mathrm{m};\ 178°;\ -20°;\ 147°\right]$
- $\mathbf{o}^* = \left[-0.39\,\mathrm{m};\ 0.33\,\mathrm{m}\right]$

Meanwhile, ${}^b\mathbf{p}_{p_0}$ and ${}^b\mathbf{p}_{p^*}$ have been measured with the HTC Vive tracking system, described in Appendix A.1.1, and serve as ground truth. It is different from the ${}^b\mathbf{p}_p$ used in the control scheme. Note that Creaform C-Track was not available to be used in these experiments.

Given the initial moving-platform pose ${}^b\mathbf{p}_{p_0}$, slack can be introduced on l_1 and l_2. In the scope of experiment the robot behavior is examined with cable slack being between 2 cm and 8 cm. More precisely, once the moving-platform is in its initial pose ${}^b\mathbf{p}_{p_0}$, the winches actuating the cables $\mathcal{C}_1$ and $\mathcal{C}_2$ were turned to increase their length by 2 cm, 4 cm, 6 cm or 8 cm. Ten experiments, named E_1 to E_{10} are defined:

- E_1 is 2½D VS without cable slack;
- E_2 is 2½D VS with TCA without cable slack;
- E_3 is 2½D VS with 2 cm cable slack;
- E_4 is 2½D VS with TCA with 2 cm cable slack;
- E_5 is 2½D VS with 4 cm cable slack;
- E_6 is 2½D VS with TCA with 4 cm cable slack;

- E_7 is 2½D VS with 6 cm cable slack;
- E_8 is 2½D VS with TCA with 6 cm cable slack;
- E_9 is 2½D VS with 8 cm cable slack;
- E_{10} is 2½D VS with TCA with 8 cm cable slack.

The experimental results are shown in Figs. 5.11 through 5.16[7]. Cable lengths, velocities and tensions are shown in Figs. 5.11, 5.12 and 5.13, respectively. Note that due to the symmetry in CDPR design and the diagonal moving-platform trajectory, only curves for cables $\mathcal{C}_1$, $\mathcal{C}_2$, $\mathcal{C}_5$ and $\mathcal{C}_6$ contain significant information. Indeed, the length and velocity curves for the other four cables are essentially the same no matter the experiment. In this first part, for a matter of clarity only experiments E_1, E_2, E_5, E_6, E_9 and E_{10} are shown. The added slack can be seen in Figs. 5.11a and 5.11b, where cable lengths l_1 and l_2 are shown.

First, the behavior of the CDPR under 2½D VS control with slack and no TCA is analyzed. Cables $\mathcal{C}_1$ and $\mathcal{C}_2$ remain longer in E_5 and E_9 than in E_1 along the whole trajectory (gray and green curves against blue curve, respectively, in Figs. 5.11a and 5.11b). Without TCA the cables $\mathcal{C}_1$ and $\mathcal{C}_2$ remain slack. However, at the end of trajectory the final cable lengths l_1 and l_2 in E_5 and E_9 are the same as in E_1. This does not mean that slack has disappeared. Indeed, it can be seen in Figs. 5.11e and 5.11f that the final length l_5 and l_6 has increased proportionally for the gray and green curves. Thus, the slack has been transferred from $\mathcal{C}_1$ and $\mathcal{C}_2$ to $\mathcal{C}_5$ and $\mathcal{C}_6$. This can also be seen in the tension measurements. Indeed, at first τ_1 and τ_2 are below the threshold τ_t, but by the end of the trajectory their values increase, while τ_5 and τ_6 decrease below τ_t. Note that more than two cables can become slack. For instance, at $t = 4\,s$ for both the gray and green curves, the cable tensions τ_2, τ_5 and τ_6 are below τ_t.

Once TCA is added to the controller, the behavior is different. It can be seen that the brown curve of E_6 aligns with pink and blue curves at approximately $t = 0.8\,s$ in Figs. 5.11a and 5.11b. Thus, at $t = 0.8\,s$ cable lengths l_1 and l_2 are equal for blue and brown curves of E_2 and E_6, respectively. Indeed, until $t = 0.8\,s$ velocity $\dot{l}_1$ of the brown curve is increased because of the TCA algorithm, as can be seen in Fig. 5.12a. Moreover, cable tension τ_1 of the brown curve also starts to increase at $t = 0.8\,s$, as shown in Fig. 5.13a. Thus, it can be concluded that 4 cm of cable slack have been successfully detected and corrected in less than one second. The cyan curves, corresponding to E_{10}, show a similar trend. Indeed, in Figs. 5.12a and 5.12b increased velocities $\dot{l}_1$ and $\dot{l}_2$ can be observed until approximately $t = 1.5\,s$. At this time the tensions τ_1 and τ_2 start to increase (see Figs. 5.13a and 5.13b) and cable lengths l_1 and l_2 align with the blue and pink curves (see Figs. 5.11a and 5.11b). It can be concluded that the larger the cable slack, the higher the time to reduce cable slack.

Note that for cables $\mathcal{C}_5$ and $\mathcal{C}_6$ the velocity curves of E_6 and E_{10} remain aligned with the curves of E_2, as can be seen in Figs. 5.12e and 5.12f. Similarly, cable length remains the

[7]Please also refer to the accompanying video at `https://youtu.be/fS8Xf3wud_4`

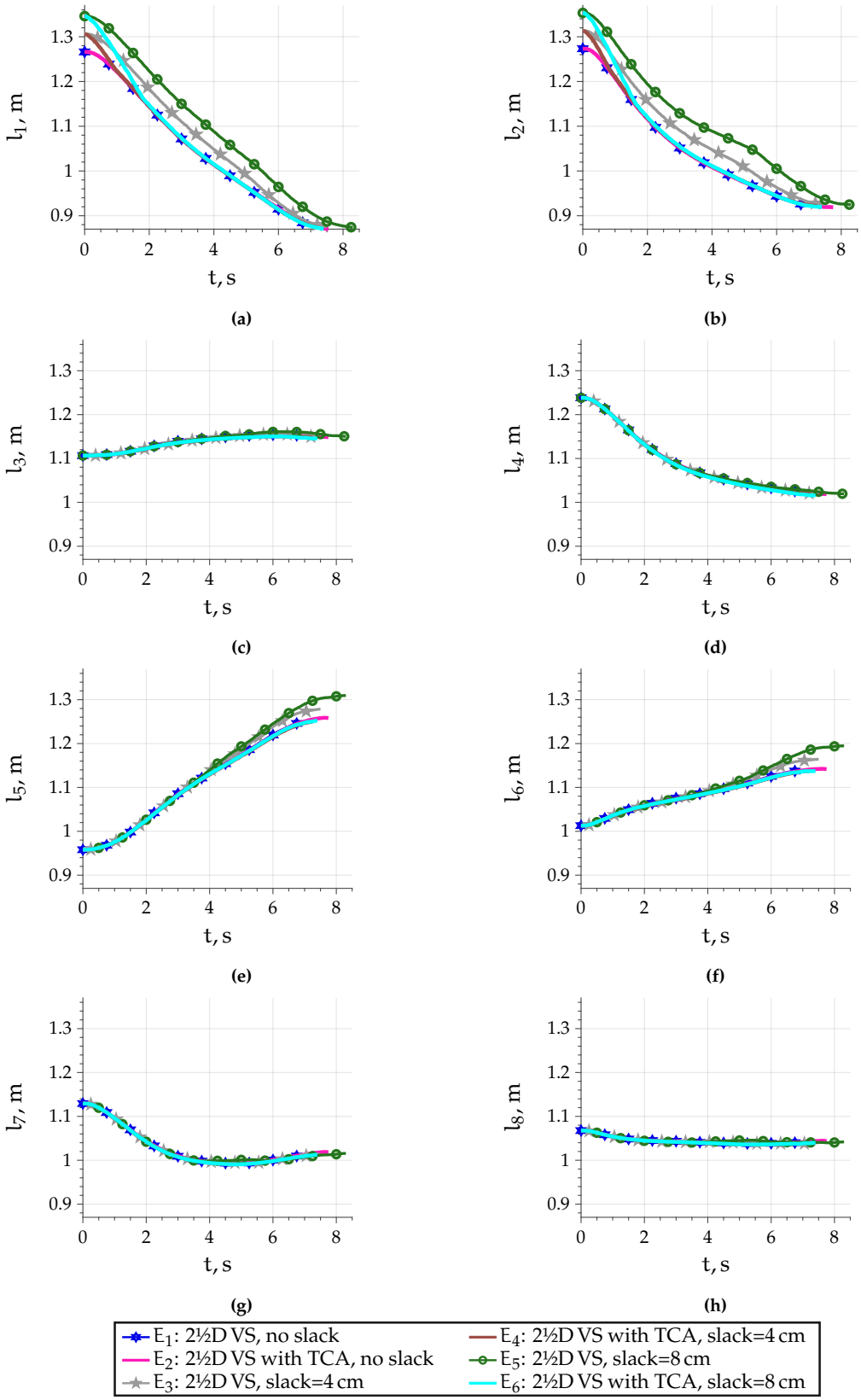


Figure 5.11: Cable lengths l_i

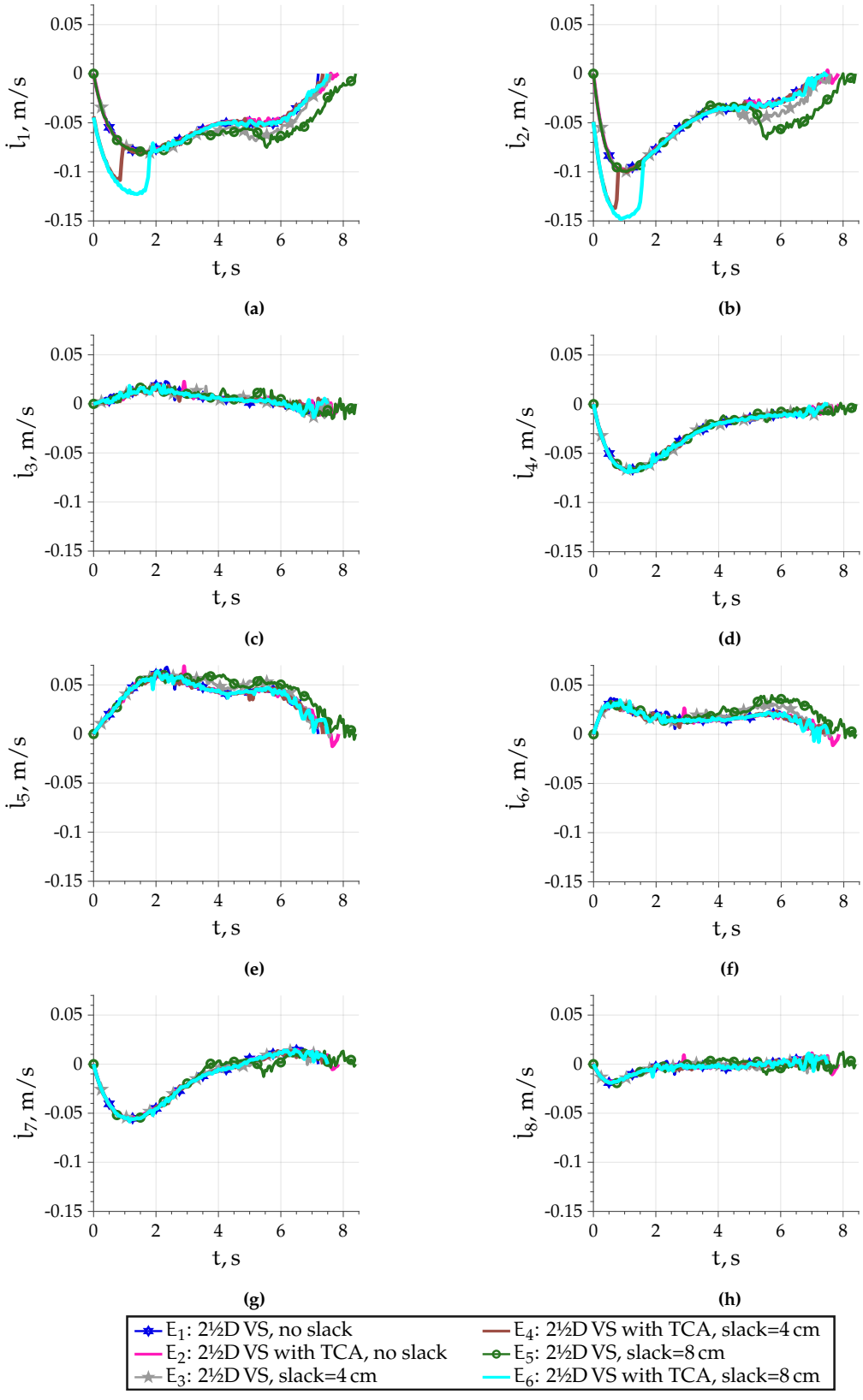


Figure 5.12: Cable velocities $\dot{l}_i$

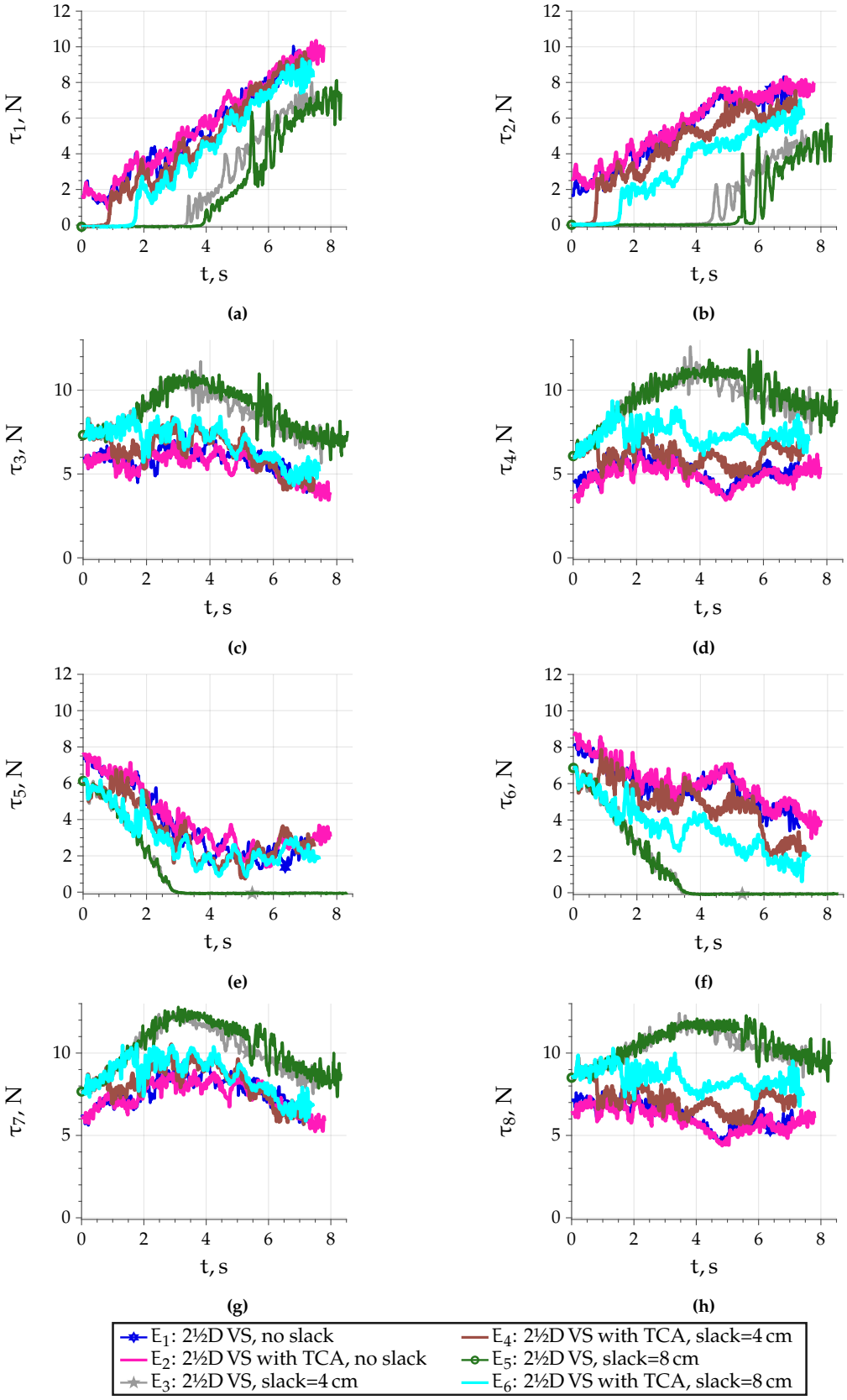


Figure 5.13: Cable tensions τ_i

same for pink, blue, brown and cyan curves in Figs. 5.11e and 5.11f, which means that no slack remains in the system. Thus, the slack is corrected before the transfer point and thus the moving-platform does not become underactuated at any point of the trajectory, when TCA is used.

Upon inspecting the final tension distribution after each task with TCA, it is clear that there are some differences. Indeed, TCA is not a TDA, it does not define precise tension values for each cable in each iteration. Instead, only a threshold is given and the tension in any cable is only corrected if it is below this threshold τ_t. For this reason, the tension distribution at the desired pose for pink, brown and cyan curves in Fig. 5.13 slightly varies. To evaluate the feasibility of the obtained tension distributions, the feasible polygon is plotted for the moving-platform at the desired pose ${}^b\mathbf{p}_{p^*}$ and is shown in Fig. 5.14. The η_1 and η_2 values corresponding to the final tension measurement of each experiment have been computed and are also shown in Fig. 5.14 and in Table 5.1. For experiments E_3, E_5, E_7 and E_9 the final tension set is not within the feasible polygon. This is because the final tension of two cables is below the threshold τ_t, which corresponds to the lower bound tension τ_{lb} that is used to delimit the feasible polygon. On the contrary, all experiments with TCA are finished with a tension distribution within the feasible polygon. However, an outward trend can be observed, where the plotted point is further from the barycenter with increased initial slack. Indeed, in E_2 no slack is added and it is the closest to the

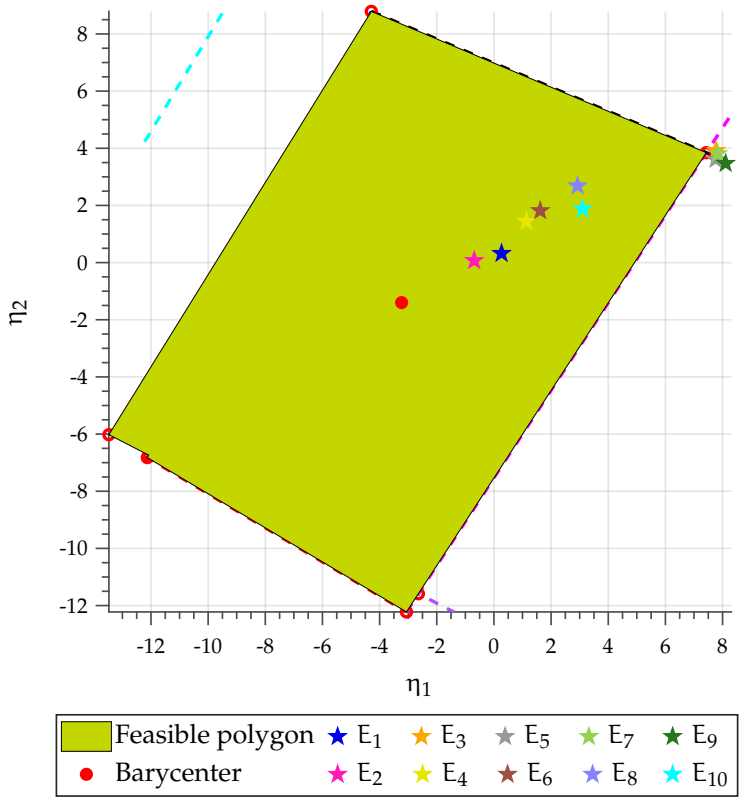


Figure 5.14: Feasible polygon of tensions at the final moving-platform pose with the final cable tension set for each experiment

barycenter. The residual is computed as $\epsilon_{Ei} = \boldsymbol{\tau}_{pv} + \mathbf{N}\boldsymbol{\eta}_{Ei} - \boldsymbol{\tau}_{Ei}$, where the particular solution $\boldsymbol{\tau}_{pv}$ is computed by (1.27). As can be seen, the residual is never zero. One of the reasons can be that it is affected by the tension sensor accuracy, which is 0.3 N. The effect of sensor accuracy can be seen in Fig. 5.15, where for each experiment the lowest and highest residual value was found within 0.3 N of the tension measurement that was recorded during experiments. Clearly, sensor accuracy has a considerable effect on the residual error. Nonetheless, with TCA it is always lower, reaching the lowest value in E_2, when no slack was added.

Table 5.1: Tension sets at final moving-platform pose and corresponding λ values

Exp.	Tension set, N	$\boldsymbol{\eta} = [\eta_1; \eta_2]$	Residual error, N
E_1	$\boldsymbol{\tau}_{E1} = [9.4; 7.3; 4.8; 6.1; 2.8; 3.7; 5.7; 5.9]$	$\boldsymbol{\eta}_{E1} = [0.27; 0.32]$	$\epsilon_{E1} = 1.21$
E_2	$\boldsymbol{\tau}_{E2} = [10.0; 7.8; 3.9; 5.4; 3.1; 4.0; 5.9; 6.1]$	$\boldsymbol{\eta}_{E2} = [-0.69; 0.07]$	$\epsilon_{E2} = 0.68$
E_3	$\boldsymbol{\tau}_{E3} = [7.2; 4.5; 7.1; 9.6; 0.0; 0.0; 8.0; 9.4]$	$\boldsymbol{\eta}_{E3} = [7.77; 3.92]$	$\epsilon_{E3} = 0.99$
E_4	$\boldsymbol{\tau}_{E4} = [9.2; 6.5; 4.6; 6.6; 2.9; 3.0; 6.2; 6.9]$	$\boldsymbol{\eta}_{E4} = [1.14; 1.44]$	$\epsilon_{E4} = 0.88$
E_5	$\boldsymbol{\tau}_{E5} = [7.2; 4.8; 7.6; 9.2; 0.0; 0.0; 7.8; 9.4]$	$\boldsymbol{\eta}_{E5} = [7.75; 3.61]$	$\epsilon_{E5} = 1.54$
E_6	$\boldsymbol{\tau}_{E6} = [9.4; 6.8; 4.8; 6.6; 2.5; 2.2; 6.4; 7.2]$	$\boldsymbol{\eta}_{E6} = [1.62; 1.82]$	$\epsilon_{E6} = 0.96$
E_7	$\boldsymbol{\tau}_{E7} = [6.8; 4.5; 7.0; 9.5; 0.0; 0.0; 8.1; 9.4]$	$\boldsymbol{\eta}_{E7} = [7.83; 3.81]$	$\epsilon_{E7} = 1.09$
E_8	$\boldsymbol{\tau}_{E8} = [9.5; 6.1; 5.3; 7.9; 2.8; 2.3; 7.5; 8.4]$	$\boldsymbol{\eta}_{E8} = [2.92; 2.68]$	$\epsilon_{E8} = 1.63$
E_9	$\boldsymbol{\tau}_{E9} = [7.4; 4.8; 7.6; 9.3; 0.0; 0.0; 8.8; 9.5]$	$\boldsymbol{\eta}_{E9} = [8.11; 3.47]$	$\epsilon_{E9} = 1.06$
E_{10}	$\boldsymbol{\tau}_{E10} = [8.3; 5.9; 5.2; 7.0; 1.8; 2.1; 6.8; 7.6]$	$\boldsymbol{\eta}_{E10} = [3.11; 1.88]$	$\epsilon_{E10} = 1.02$

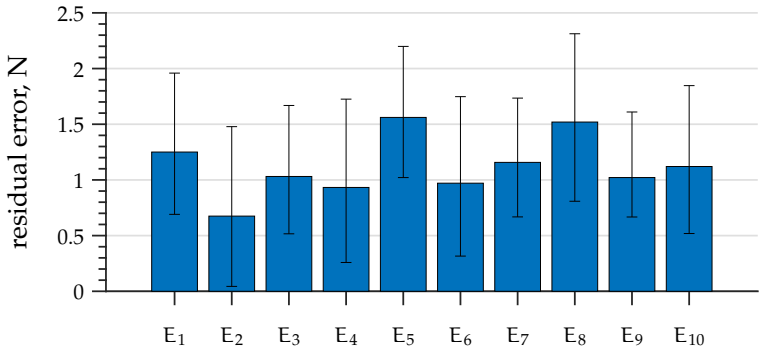


Figure 5.15: Residual error as a function of tension sensor accuracy

The effect of cable slackness on the AprilTag center-point trajectory in the image is visible in Fig. 5.16. Here, all ten experiments are shown, including E_3, E_4, E_7 and E_8. The larger the initial cable slack, the larger the deviation from the straight-line trajectory. As the moving-platform rapidly falls to a new cable configuration, it causes a sideways drift in the image trajectory. In fact, the cable configuration is changed twice in the middle of the trajectory, which is the reason why the trajectory in Fig. 5.16a exhibits a drift to both

sides for E_5, E_7 and E_9. This is also the reason why there are two spikes on the trajectory deviation plot in Fig. 5.16b. Furthermore, right after the second spike a rapidly damping oscillation can be seen for E_5, E_7 and E_9. This corresponds to the rapid changes of τ_1 and τ_2 in Figs. 5.13a and 5.13b. Indeed, as the moving-platform is not stiff and the cables are very slack, the moving-platform oscillates slightly due to the inertia of its fall to a different cable configuration.

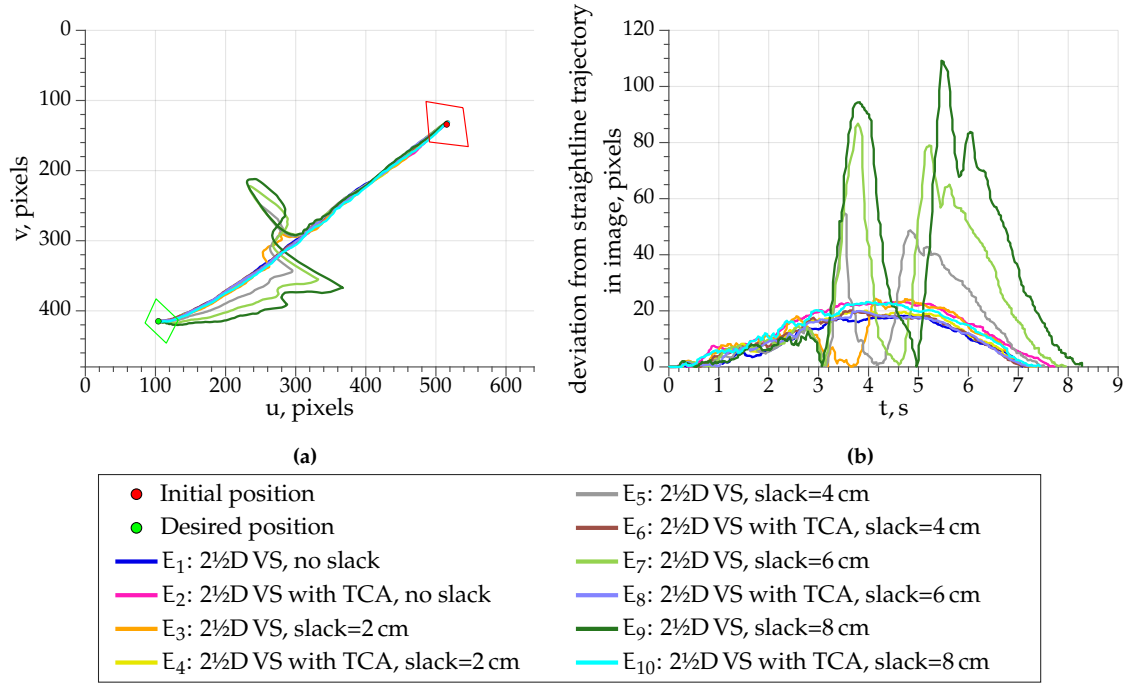


Figure 5.16: AprilTag center-point trajectory in the image and the deviation from the desired straight-line trajectory in pixels

On the contrary, when TCA is used, the produced trajectories are very close to the desired straight lines. Indeed, the deviation with TCA is approximately the same in every experiment. Thus, TCA does not perturb the main controller. Indeed, it improves the visual servoing controller by reducing cable slack. However, it does not improve the visual servoing controller's robustness to other perturbations. To reduce this deviation, one could combine the trajectory planning and tracking with TCA.

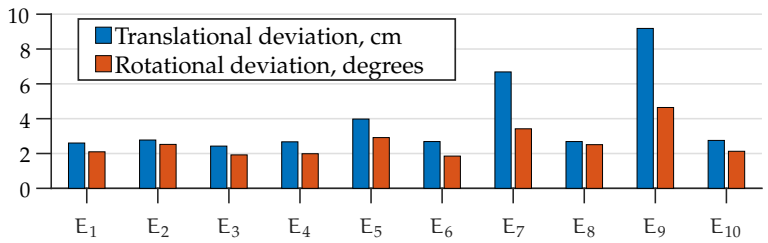


Figure 5.17: Translational and rotational deviation of final moving-platform pose estimation

Eventually, cable slackness affects the accuracy of moving-platform pose estimation, because slack cables do not produce the desired moving-platform displacement. In

Fig. 5.17 the distance between the actual final moving-platform pose and its estimation is shown for each of the ten experiments. Here, the rotational deviation is defined as the angle θ of the axis-angle $\theta\mathbf{u}$ representation of the rotation matrix ${}^{\hat{p}}\mathbf{R}_p$ between the estimated and the actual moving-platform pose. In all five experiments, where TCA is used, the deviation remains approximately the same, that is, the translational error is approximately 0.03 m and the rotational error is approximately 2°. On the contrary, when TCA is not used, the estimation deviates reaching almost 10 cm and 5° error for E_9. Given the short trajectory time of only 8 s, it can be concluded that large cable slack leads to fast deviation of the moving-platform pose estimation. Moreover, the larger the initial cable slack, the larger the error in moving-platform pose estimation.

Stability Analysis of the Cable Slackness Experiments

This section deals with the stability analysis for the given experiments. The stability criterion $\boldsymbol{\Pi}$ is computed off-line as defined in (3.1) and the recorded variables were used as follows:

- the same in the model and estimation: $\mathbf{s}$, $\mathbf{s}^*$, $\mathbf{L}_s$, A_i, B_i, ${}^p\mathbf{T}_c$, $\mathbf{A}_d$;
- estimation: ${}^b\widehat{\mathbf{T}}_p$ as estimated by the controller;
- model: ${}^b\mathbf{T}_p$ acquired from HTC Vive tracking system.

The chosen expression of the CDPR model and thus stability criterion $\boldsymbol{\Pi}$ does not contain cable tensions. For CDPRs with light cables, if one cable becomes slack, as detected by tension measurements, then it can be considered non-existent. Indeed, a light cable exerts a negligible force on the moving-platform when slack. Furthermore, while a cable is slack, reducing its length without elimination of slack does not induce any moving-platform motion. The expression of the Jacobian matrix $\boldsymbol{A}$ is updated at every iteration accordingly. For example, if at $t = 0$ s tension $\tau_{m_1} < \tau_t$, then the corresponding row of matrix $\boldsymbol{A}$ is removed and it becomes a (7×6)–matrix. Furthermore, if more than two cables are slack, the Jacobian $\boldsymbol{A}$ will be rank deficient and the robot will be underactuated. Considering the stability criterion $\boldsymbol{\Pi}$, it is clear that once $\text{rank}(\boldsymbol{A}) < 6$, then $\text{rank}(\boldsymbol{\Pi}) < 6$ and thus the stability criterion will not be fulfilled.

Fig. 5.18 contains the results of stability analysis over time. Here, the Boolean *Stable* is true if $\boldsymbol{\Pi} > \mathbf{0}$ and false otherwise. As expected, when no cable slack is added to the system, it remains stable throughout the trajectory no matter the controller, as shown in Fig. 5.18a. It can also be seen that at one iteration one tension measurement goes below the threshold. However, that does not render the system unstable, because there are still seven cables in tension. For experiment E_3, shown in Fig. 5.18b in orange, in the middle part of the trajectory three cables are becoming slack. Furthermore, at some iterations even four cables are becoming slack. As the cable slack is increased in experiments, the time period with moving-platform being underactuated also increases. Indeed, in E_3, shown in Fig. 5.18b, the moving-platform is underactuated for approximately two

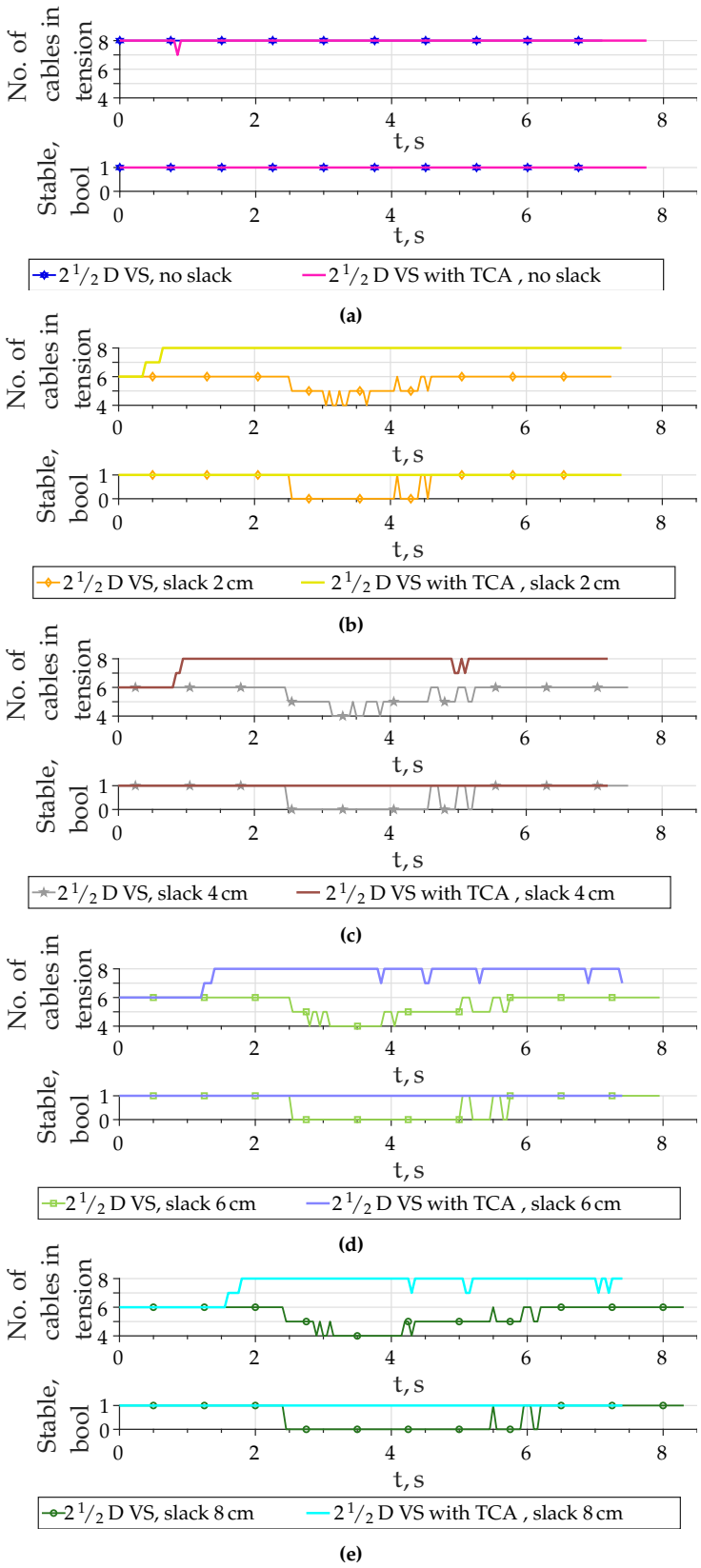


Figure 5.18: The correlation between the amount of cables in tension and system stability for: (a) no slack; (b) 2 cm slack; (c) 4 cm slack; (d) 6 cm slack; (e) 8 cm slack

seconds. However, in E_9, shown in Fig. 5.18e, the moving-platform is underactuated for approximately four seconds, which is half of the trajectory time.

Once the moving-platform is underactuated, $\mathbf{\Pi} > \mathbf{0}$ is no longer true. Thus, when the moving-platform is underactuated, it is no longer in equilibrium and the stability of the system is not ensured. Hence, the equilibrium of the moving-platform and the stability of the system are linked.

On the other hand, when TCA is added to the control, it rapidly corrects the slack and the system remains stable all along the trajectory, as can be seen in Fig. 5.18. In this figure one can clearly see that by increasing the cable slack, the time required to fix it increases proportionally. For example, in E_4 with 2 cm slack, shown in Fig. 5.18b, 0.4 s are needed for the first slack cable to become tensed. In E_6 the slack is doubled, i.e. it is now 4 cm and the required time to fix it is also doubled to 0.8 s. And finally in E_{10} the slack is doubled once more and the required time to fix it is approximately 1.6 s.

It can be concluded that if the CDPR becomes underactuated due to cable slack, it makes the moving-platform lose its equilibrium and the stability criterion $\mathbf{\Pi} > \mathbf{0}$ is no longer ensured.

Tracking a Mobile Robot

In this subsection the CDPR behavior over a longer time period is shown. ACROBOT is tasked with tracking a small mobile robot that itself is following a black line as shown in Fig. 2.15. A PBVS controller is used and it has no knowledge of the mobile robot trajectory. It simply attempts to keep the AprilTag of the mobile robot in the desired pose. At the end of every lap, a known AprilTag comes in the field of view. This allows the controller to reinitialize the moving-platform pose estimation. As shown in Section 2.3.7, without this cheat the tracking fails at $t = 210$ s.

The moving-platform trajectory is shown in Figs. 5.19 and 5.20 for PBVS without and with TCA, respectively. The cable length sum change over different laps is shown in Fig. 5.21. Finally, the tension measurements are shown in Figs. 5.22 and 5.23 for PBVS without and with TCA, respectively.

As can be seen in Fig. 5.19, the classic PBVS fails in lap 29, making the total task execution time just under ten minutes. Compared to results of Section 2.3.7, it is clear that using a known tag to reinitialize the moving-platform pose estimation is beneficial. Indeed, the total time has been tripled. Nevertheless, the moving-platform trajectory gradually becomes worse. Note that the deterioration of the trajectory changes exponentially. For example, the deterioration between laps 10 and 20 is comparable to the one between laps 20 and 24. Sharp deviations can be seen on the farthest corners of the track. These are due to slack in the cables. Thus, even though moving-platform pose estimation is frequently reinitialized, the task still fails.

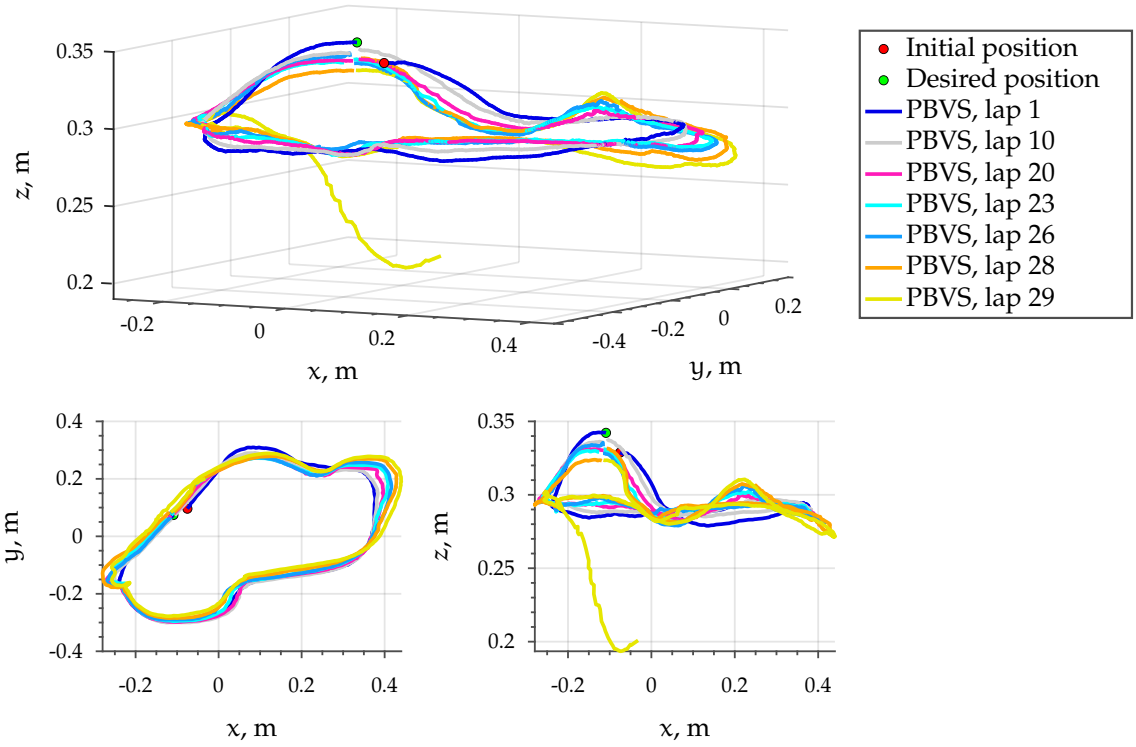


Figure 5.19: Moving-platform trajectory with classic PBVS

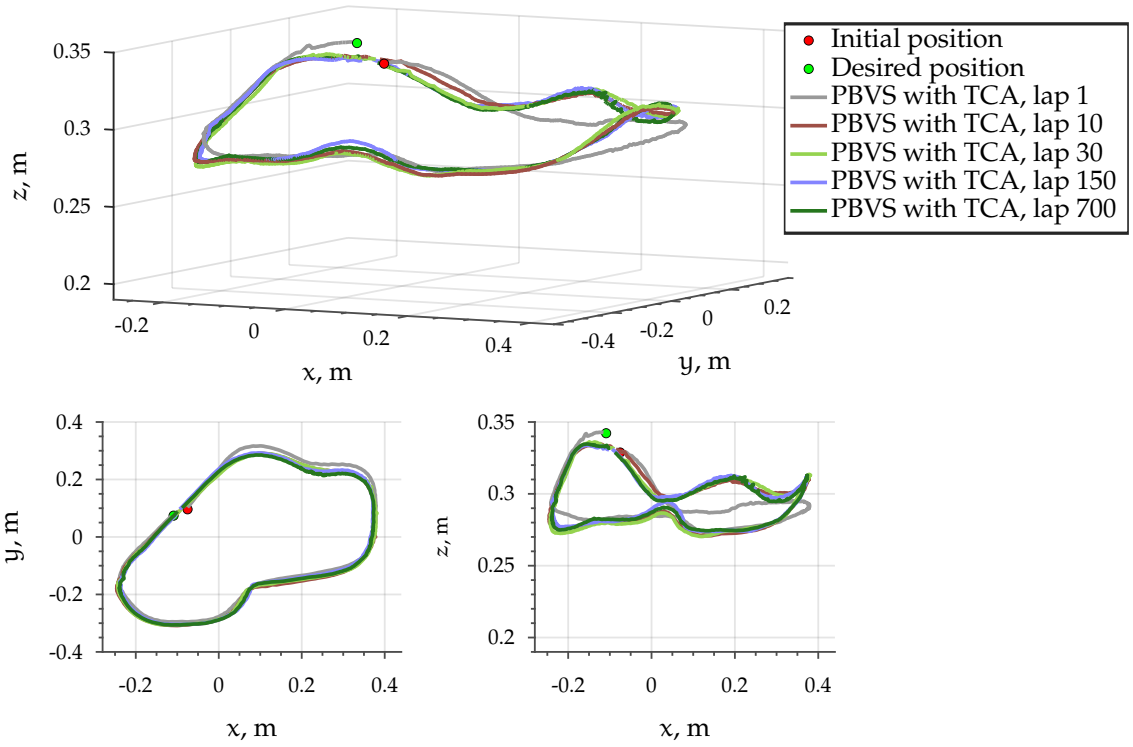


Figure 5.20: Moving-platform trajectory with PBVS with TCA

On the other hand, the moving-platform trajectory of the VS with TCA, shown in Fig. 5.20, remains almost exactly the same be it lap 10 or 700. Indeed, only the first lap,

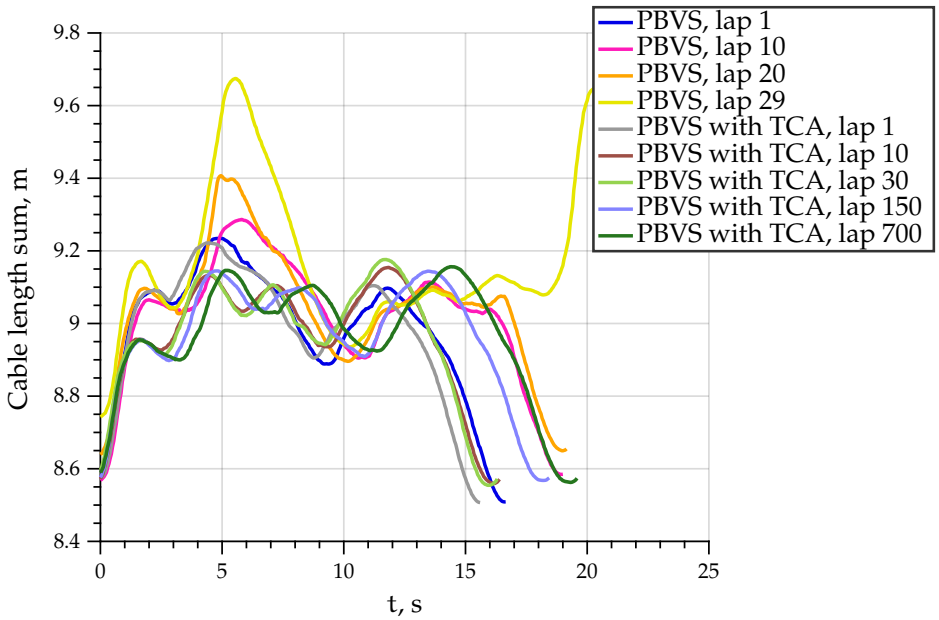


Figure 5.21: Sum of cable lengths over time

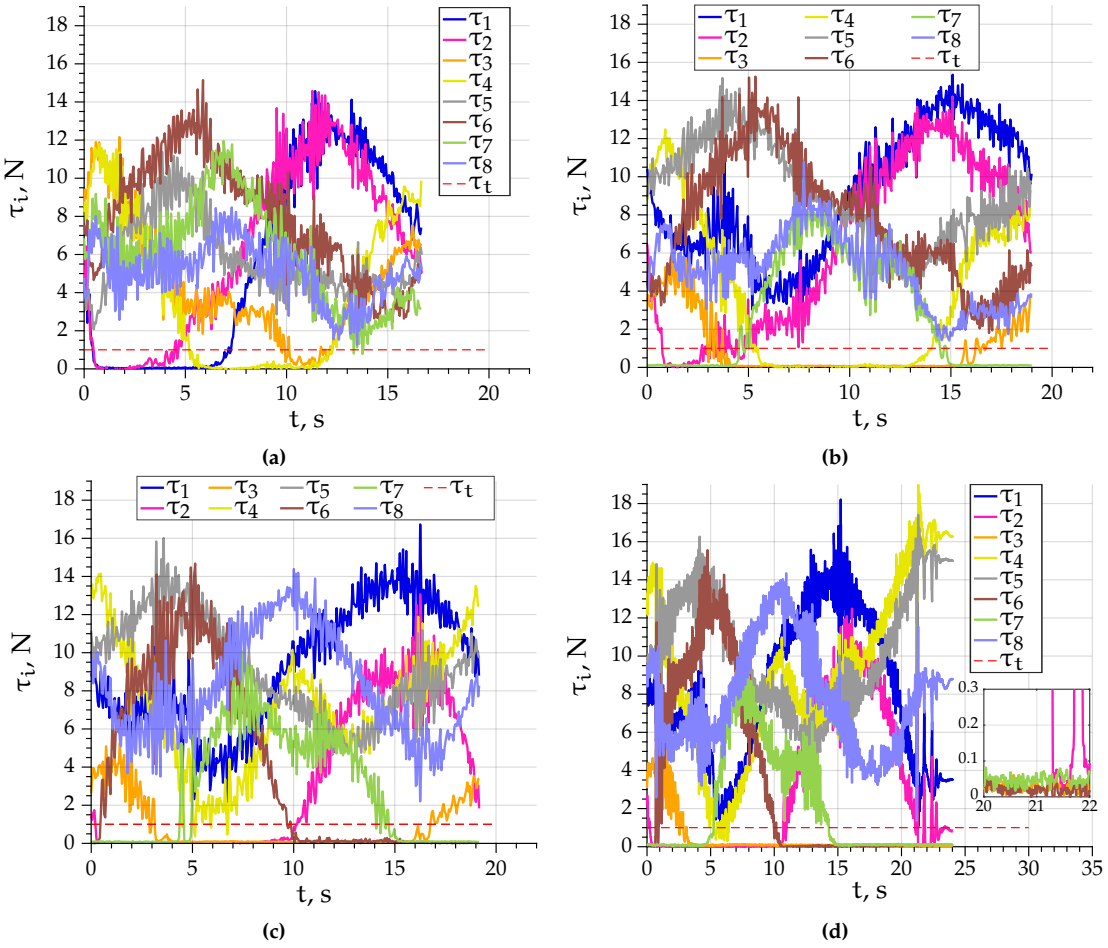


Figure 5.22: Tension measurements with classic PBVS: (a) lap 1; (b) lap 10; (c) lap 20; (d) lap 29

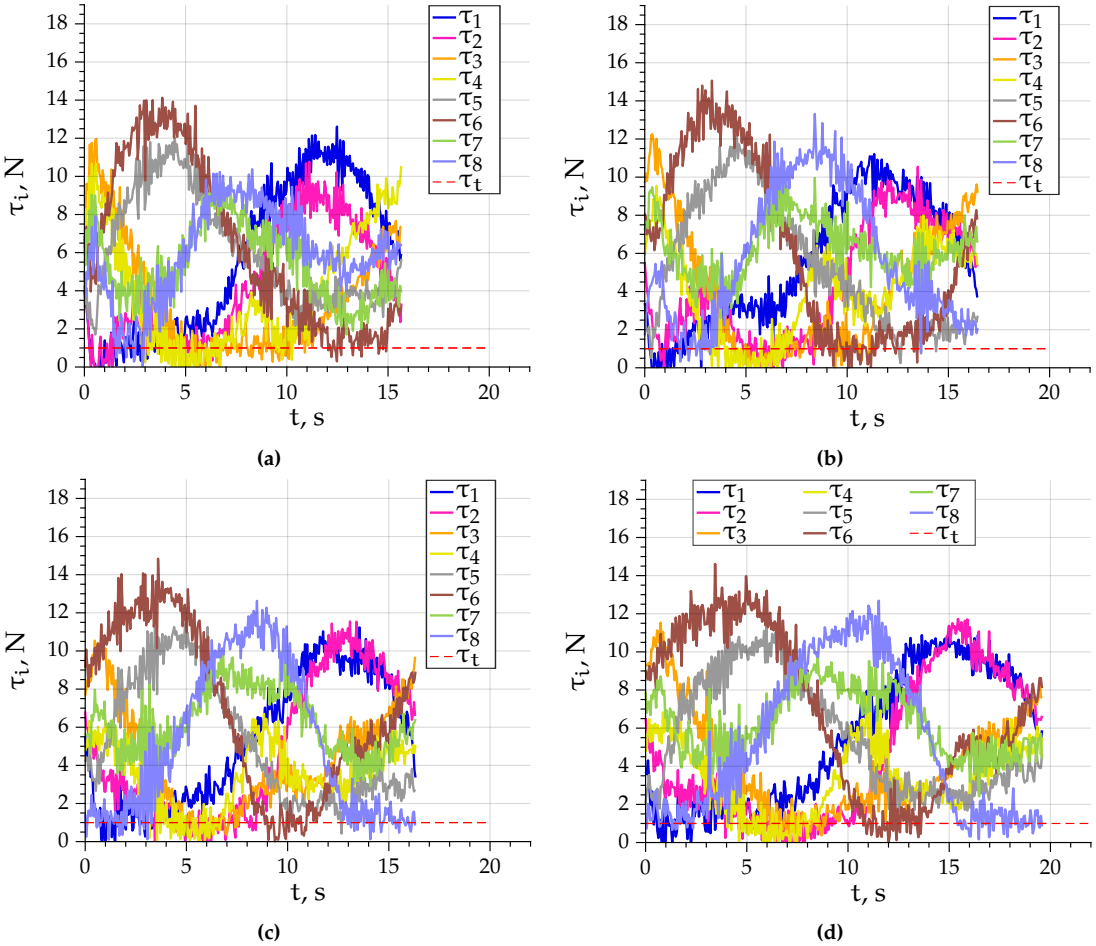


Figure 5.23: Tension measurements with PBVS and TCA: (a) lap 1; (b) lap 10; (c) lap 30; (d) lap 700

shown in gray, is significantly different. However, this is due to the manual placing of the mobile robot on the track and initialization of the task. This is also true for the experiment without TCA, where the first lap is shown in dark blue in Fig. 5.19.

It can be concluded that, when TCA is used, the quality of tracking the mobile robot in lap 700, which occurred after 180 minutes, remains as good as in the beginning of the task. The experiment was stopped, because no deterioration of behavior could be seen.

The cable length sum is shown in Fig. 5.21. It allows us to rapidly detect any significant increase of cable slack in the system, regardless of individual cable length changes. Here, it can be seen that a noticeable increase in cable lengths appears already in lap 10 for classic PBVS, shown in pink. Then in lap 20, shown in orange, at some instants it reaches approximately 0.15 m compared to lap 10. Finally, in lap 29, shown in yellow, the spike at around $t = 5\,s$ surpasses the curve of lap 20 by additional 0.3 m, which eventually leads to task failure at the beginning of the next lap. On the contrary, when TCA is used, cables are kept in tension and do not become slack.

In Fig. 5.22a it can be seen that even during the very first lap, most of the time one cable was slack, sometimes even two cables. However, during lap 1 the slack is insignificant

and it transfers rapidly to other cables. During laps 10 and 20 two cables are always slack. The transfer of slack takes up to 1 s in lap 20, as shown in Fig. 5.22c at $t = 15\,s$. Here, $\mathcal{C}_7$ becomes slack, when there are already two cables, namely $\mathcal{C}_3$ and $\mathcal{C}_6$, that are slack. Cable $\mathcal{C}_3$ stops being slack only after $t = 16\,s$. Thus, the moving-platform was underactuated during one second.

The situation is significantly worse during lap 29. As can be seen in the small zoomed segment in Fig. 5.22d, during task failure at $t = 22\,s$ there are already three cables slack and a fourth one becomes slack momentarily. More precisely, cables $\mathcal{C}_3$, $\mathcal{C}_6$ and $\mathcal{C}_7$ are all slack in the time period between $t = 15\,s$ and task failure. Thus, the last six seconds the moving-platform was underactuated, which lead to the task failure.

On the contrary, when TCA is used, no more than two cables are slack at any given time, as can be seen in Fig. 5.23. It can also be seen that TCA is participating in the control throughout each lap. Indeed, there are usually one or two cable tensions that are right on the threshold τ_t. There is no significant difference between tension curves in lap 10, 30 or 700, shown in Figs. 5.23b, 5.23c and 5.23d, respectively. Indeed, it appears that no matter whether two or 180 minutes have passed since the beginning of the task, the behavior is similar. It can thus be concluded that TCA successfully deals with cable slackness and keeps the CDPR responsive to the control signals of PBVS.

5.3.2 Case Study X: VS with TCA on CAROCA

Experimental Setup

The ten experiments with increasing amount of slack were repeated on the large CDPR as well. The initial values are the following:

- ${}^b\mathbf{p}_{p0} = \left[-0.608\,m;\ -0.925\,m;\ 1.501\,m;\ 3°;\ 0°;\ -23°\right]$
- ${}^c\mathbf{p}_{o0} = \left[0.547\,m;\ -0.287\,m;\ 1.339\,m;\ 179°;\ 2°;\ -160°\right]$
- $\mathbf{o}_0 = \left[0.39\,m;\ -0.21\,m\right]$

and desired values are selected to be:

- ${}^b\mathbf{p}_{p^*} = \left[0.873\,m;\ 0.308\,m;\ 1.668\,m;\ -16°;\ 4°;\ -2°\right]$
- ${}^c\mathbf{p}^*_o = \left[-0.698\,m;\ 0.586\,m;\ 1.764\,m;\ -161°;\ -4°;\ -28°\right]$
- $\mathbf{o}^* = \left[-0.39\,m;\ 0.33\,m\right]$

2½D VS is used as the main controller. The tuning of TCA is more complex on CAROCA. This is because it is equipped with tension sensors that are made for high loads up to 25000 N. The accuracy of these sensors is 0.3% of the full scale output or 75 N. Furthermore, due to sensor calibration and also due to cable weight, the sensors do not show 0 N when their cables become slack. Indeed, for the moving-platform of approximately 150 kg the tension measurement with slack cables at the center of the workspace is: $\boldsymbol{\tau}_s = \left[120\,N,\ 90\,N,\ 90\,N,\ 50\,N,\ 90\,N,\ 100\,N,\ 0\,N,\ -20\,N\right]$. As shown in Fig. 1.18, setting tension lower bound $\tau_{lb} = 30\,N$ does not reduce the SFW too much. Thus, for CAROCA we set $\tau_t = \tau_{lb} = 30\,N$. Note that this value needs to be added to $\boldsymbol{\tau}_s$ in order to get the correct

threshold tensions: $\boldsymbol{\tau}_t = \boldsymbol{\tau}_s + \tau_t = \big[150\,\text{N}, 120\,\text{N}, 120\,\text{N}, 80\,\text{N}, 120\,\text{N}, 130\,\text{N}, 30\,\text{N}, 10\,\text{N}\big]$. The issue with sensor accuracy is that for different moving-platform poses and thus different cable lengths, the sensor will register a substantially different slack tension.

Due to increased tensions, the gain k_c needs to be tuned to a significantly smaller value. Finally, it was set to $k_c = 0.001\,\text{m/Ns}$.

Experimental Results

The results are shown in Figs. 5.24 and 5.25. Note that due to the good tensioning for the initial moving-platform pose, once slack is added in the system, the moving-platform rotates a little, which creates a small displacement of the initial AprilTag center-point, as can be seen in Fig. 5.24a. Of course this displacement does not affect the controller, because the current AprilTag pose is computed at every iteration and the desired pose has been left unchanged.

When no slack exists, the trajectories produced by 2½D VS with and without TCA are almost identical as shown in Fig. 5.24 with pink and blue curves, respectively. Then, as the slack increases, the deviation of the AprilTag center-point trajectory becomes larger for 2½D VS. This is also true for the camera trajectory in the base frame. Indeed, for 6 cm slack, shown in light green, the deviation reaches 80 pixels and 100 mm. Then, with the slack increased to 8 cm the task fails (shown in dark green).

On the other hand, when using TCA slack is rapidly corrected. For 6 cm slack, shown in violet, the deviation reaches only 35 pixels and 20 mm. Only the reduction of 8 cm slack takes enough time to produce significant deviation in the trajectories as it reaches 90 pixels and 40 mm (shown in cyan).

As can be seen in Fig. 5.25, the tension measurement range for CAROCA is significantly larger and for some cables such as $\mathcal{C}_1$ the measurement never reaches the threshold. Furthermore, as mentioned before, the difference between the initial tensions is significant when comparing the experiments with and without slack. Due to adding slack, τ_2 and τ_7 are just below their corresponding threshold values at $t = 0\,\text{s}$. Then, $\mathcal{C}_7$ remains slack for approximately 10 s for the experiments without TCA, namely E_3, E_5, E_7 and E_9. On the contrary, for the experiments with TCA the tension measurement τ_7 reaches its threshold in less than 2 s. Around $t = 4\,\text{s}$ the tension measurement τ_3 reaches the threshold. Then the curves corresponding to experiments without TCA descend below the threshold, while the rest remains on the threshold. Note that during the trajectory for some experiments τ_4 and τ_6 also reach the threshold, where similarly the TCA ensures that the tension measurement does not pass below the threshold. Overall, TCA was necessary during most of the trajectory, despite the good trajectories produced by the controller no matter cable slack (with the exception of E_{10}).

The final moving-platform pose estimation deviation from the Creaform C-Track measurement is shown in Fig. 5.26. To make these measurements C-Track was positioned as shown in Fig. A.12d of Appendix A.2 in order to cover the workspace as much as

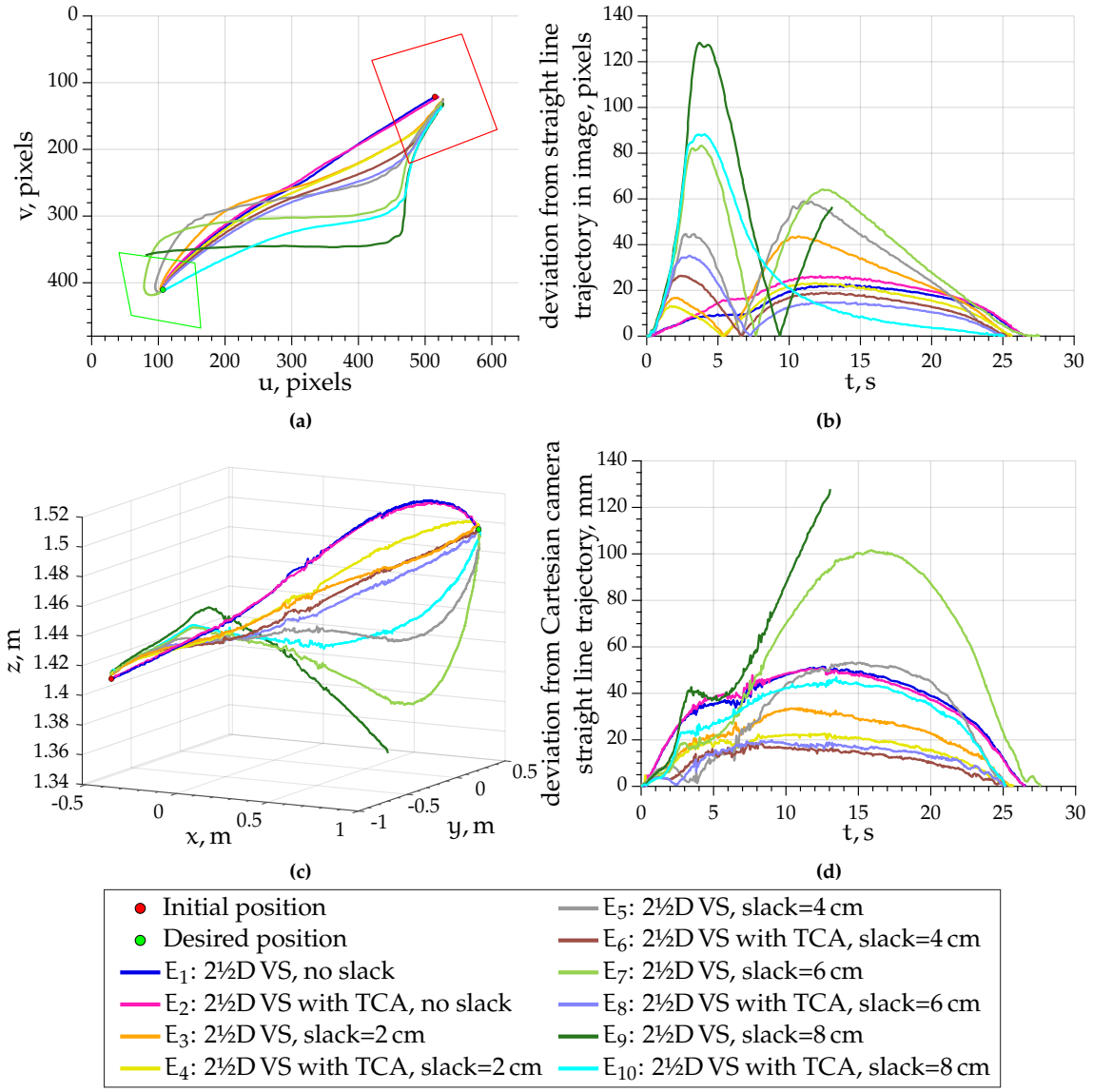


Figure 5.24: **(a)** AprilTag center-point trajectory in the image and **(b)** the deviation from the desired straight-line trajectory in pixels; **(c)** camera trajectory in the base frame $\mathcal{F}_b$ and **(d)** the deviation from the desired straight-line trajectory in mm.

possible. The deviation at the end of experiments without slack, namely E_1 and E_2 is the same, meaning that TCA does not intervene with moving-platform pose estimation. Indeed, it only pulls on the cables that are too slack to participate in the control and thus does not induce any undesirable motion. When slack exists in the system, deviation remains approximately the same for E_4, E_6 and E_8, while it becomes worse for E_{10}. This is because in E_{10} the 8 cm slack was not fully corrected when the slack cable became necessary and thus some limited undesired motion was produced by the moving-platform changing its six-cable configuration.

For the given experiment, adding slack appears to affect the orientation estimation significantly more than the position estimation. Indeed, in E_3, E_5 and E_7 the rotational

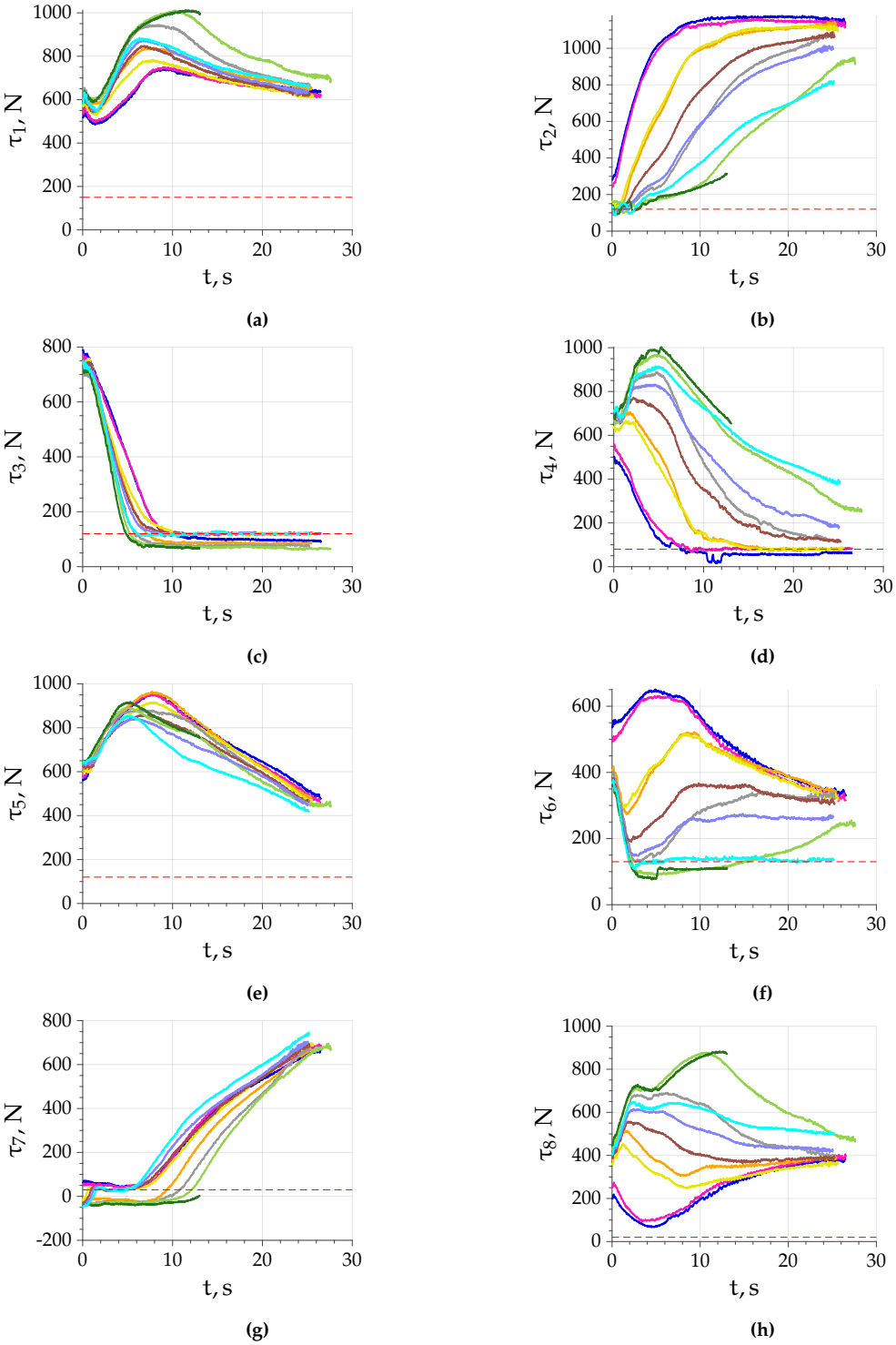


Figure 5.25: Cable tensions τ_i

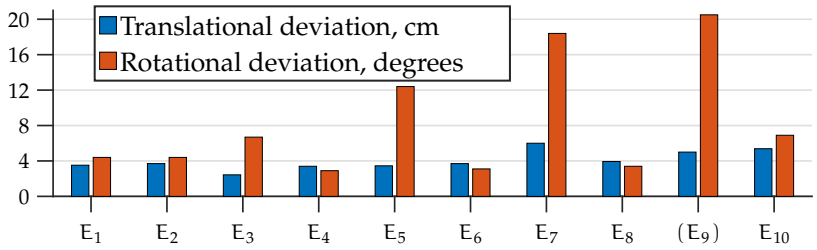


Figure 5.26: Translational and rotational deviation of final moving-platform pose estimation

deviation grows rapidly. On the other hand, the translational deviation at the end of E_3 is actually smaller than in the other experiments. However, it does increase with the increase of added slack, reaching 6 cm in E_7. Note that for E_9 we show the deviation at the moment of task failure. The translational deviation had already reached 5 cm while the rotational deviation had surpassed 20°.

5.4 Conclusions

In this chapter, multiple tension management approaches were proposed. The approaches can be distinguished by the different level of abstraction from visual servoing of CDPRs as it was introduced in Chapter 1. In the simplest case, the visual servoing control scheme is simply enriched by a tension correction algorithm (TCA). In the TCA the current tensions are compared to a minimum threshold and if they are too low, a cable velocity correction is computed. On the other hand, in the most changed approach the current object and the current moving-platform poses are obtained through computer vision. Then , knowing the desired object pose, it is possible to compute the corresponding moving-platform pose and thus control cable lengths or tensions to ensure that the current moving-platform pose converges to the desired one.

The TCA was then implemented on the two CDPRs to deal with cable slackness. It is a simple algorithm that is efficient at its task of reducing cable slack, while not perturbing the main controller. Indeed, when simplicity is efficient, it can be considered as an advantage. The tension correction only occurs, when cable slackness is detected and it is stopped as soon as the cable is tensed enough to pull on the moving-platform. The proposed algorithm is efficient in reducing cable slack and its rapidity depends on the tuning of the gain and the amount of slack on the cable.

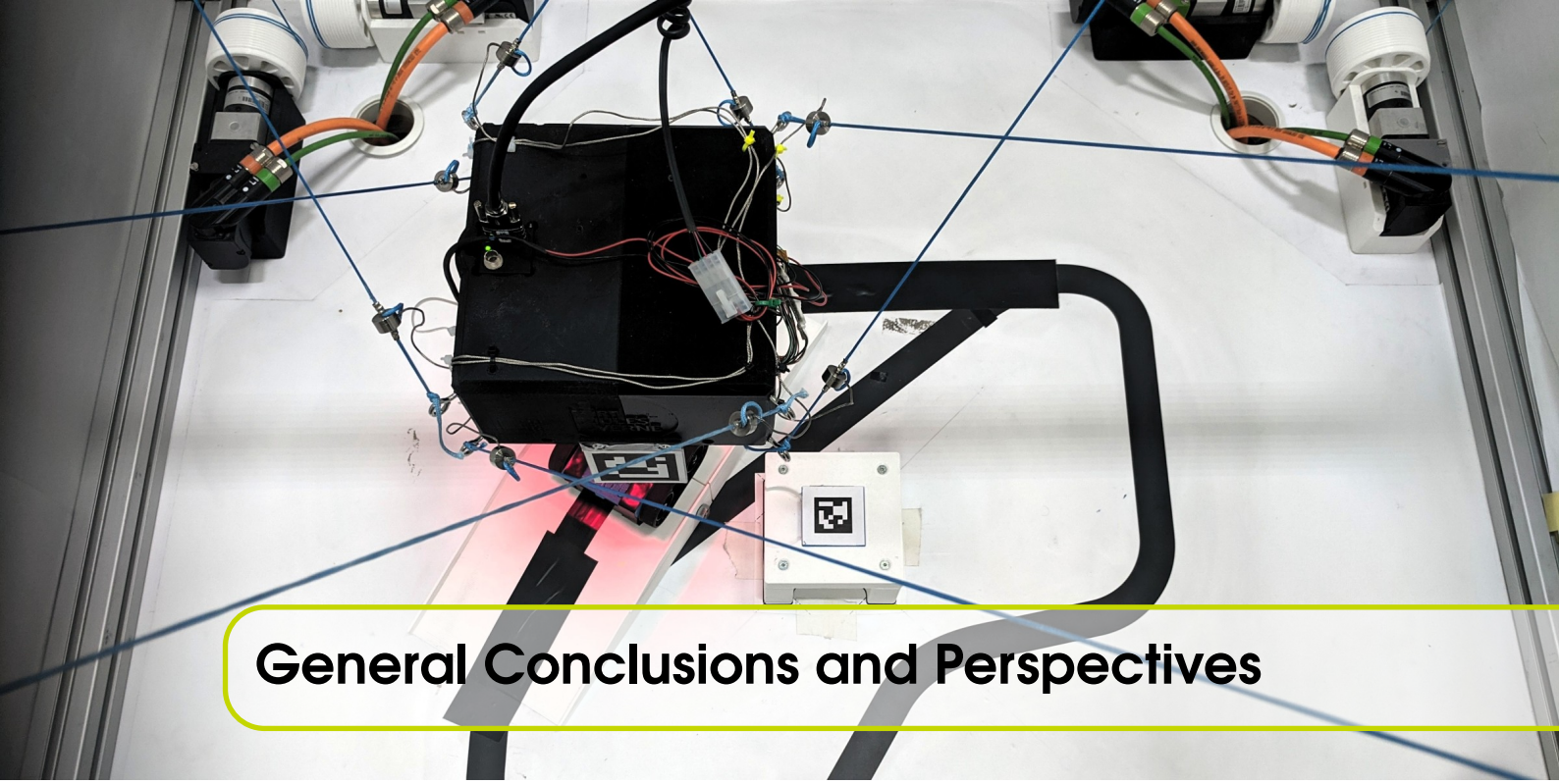
Cable slack affects not only the CDPR responsiveness to control output, but also its stability. Indeed, when slack is large, the moving-platform can become underactuated along the trajectory. Furthermore, underactuation leads to the stability criterion being no longer ensured. In addition, the larger the cable slack, the larger the moving-platform pose estimation error, when TCA is not used.

When cable slack is transferred between cables, the moving-platform can sway uncontrollably, leading to sharp changes of the target trajectory in the image. As the

cable slack increases, it becomes more likely that the target will leave the field of view, resulting in task failure.

Cable slack accumulates rapidly in longer tasks. This is true even if a periodical moving-platform pose update is available. On the contrary, using a tension correction algorithm along with a periodical moving-platform pose update allows the CDPR to execute prolonged tasks without any significant deterioration of system behavior.

Tension correction only removes cable slack, it does not address the underlying causes. TCA greatly improves the responsiveness of the CDPR to the control signal. However, other perturbations can exist in the system and the produced trajectory can be not ideal. In this case a combination of TCA and trajectory planning and tracking could be used.



General Conclusions and Perspectives

General Conclusions

This thesis was focused on the implementation and analysis of visual servoing for CDPRs in order to improve their accuracy. The advantage of visual servoing is that it is possible to continuously observe an object of interest and thus to know when the desired state has been reached. Thanks to this, the accuracy at the goal state is always the same. Indeed, no matter the perturbation, as long as the system converges the desired state is achieved with the same accuracy. Furthermore, since an onboard camera is used for the visual servoing, it approaches the object together with the moving-platform, and the closer it is to the object the higher the accuracy with respect to the object.

Three visual servoing approaches were analyzed: image-based, pose-based and 2½D visual servoing. It was rapidly found that a CDPR with visual servo control is robust to many perturbations, for example having only a coarse estimation of the initial moving-platform pose. To understand the limits of this robustness an extensive Lyapunov stability analysis was performed first on a simple planar CDPR and then on a spatial one. It was concluded that many parameters can be perturbed in the system and all these perturbations have a joined effect on system stability. Indeed, for an ideal robot it is possible to voluntarily increase the perturbation on one parameter to a very high value. However, once other perturbations are taken into account, then each of them can only be increased to a certain limit before making the system unstable.

Furthermore, a link between the moving-platform pose and the stability of the system was determined. That is, if in one pose a certain set of perturbations does not make the system unstable, then in a different pose the same set can make the system unstable. Thus,

a novel workspace, named Control Stability Workspace, was defined. It is the set of all moving-platform poses for which the stability criterion is held as long as the perturbation set is within the predefined perturbation bounds. As with all workspaces, CSW is model-dependent, thus it had to be computed separately for the two spatial CDPRs that were available for experimental validation. Moreover, CSW is also control-dependent, so it was computed for each of the visual servoing approaches. The effect of each perturbation on the system stability can be evaluated through CSW. More precisely, by increasing the amplitude of a perturbation the CSW volume decreases, however the rate is different based on which parameter is being perturbed. For example it was found that camera position in the moving-platform can be very coarsely set. That is, some arbitrary poses within the moving-platform can be given to the controller and that will have no effect on the workspace size. On the other hand the perturbation range on cable exit and anchor points is directly related to the size of the CDPR prototype. That is, the larger the base frame of the CDPR, the larger the perturbation on cable exit points within system stability. Similarly, the larger the moving-platform, the larger the perturbation on cable anchor points within system stability. For ACROBOT the pulleys and anchor mechanisms are so small that they can be assumed to be points without greatly reducing the size of CSW. For CAROCA the pulleys are significantly larger, thus the CSW is reduced by not using pulley geometry in control, however the remaining workspace is still sufficiently large.

In order to ensure the best accuracy, it was chosen to mount the camera on the moving-platform in the eye-in-hand configuration. However, in this configuration the moving-platform itself is not observed and thus its pose cannot be directly measured. Instead it needs to be estimated. Three different estimation approaches were developed: control-based, image-based, and model-based. The control-based approach is based on the integration of control output. The image-based approach is based on the use of two images to recover two object poses and thus compute the new moving-platform pose from the transformation of frames. Note that one can either use two consecutive images or the first and the current image for this approach. Finally, the model-based approach signifies the computation of the moving-platform pose from cable lengths, which is basically solving the direct geometric model. Here, four CDPR models were used: the simplest model; model taking into account cable elasticity; model taking into account pulley geometry; and model taking into account both cable elasticity and pulley geometry. Note that tension sensors are used to find the six cables that are most in tension and only these six cables are used in the computation of the moving-platform pose. Thus, altogether seven estimation methods were tested on two different CDPRs and with the object being either static or in motion. It was found that control integration is the most robust moving-platform pose estimation method. With the latter the robot produced the best trajectories in the presence of perturbations. Furthermore, using control integration lead to the longest mobile object tracking time before failure. The drawback of image-based methods is that a static object has to be observed, while in reality it can also move, be it on purpose or by accident. It also

has to appear large in the image to provide a good accuracy of the estimated pose. The drawback of model-based methods is that they require the robot to correspond well to the model and cable slackness leads to finding the wrong moving-platform pose. Furthermore, these methods rely on the use of tension sensors that may not be available. Interestingly, it was found that the smoothness of estimation is more important than estimation accuracy. Control integration provides a very smooth change of moving-platform pose that can deviate significantly over an increased period of time. On the other hand model-based methods are very precise with the exception of some wrong computations due to cable slackness. While control integration was found to be the most suitable for our needs, it also has drawbacks. More precisely, it relies on the assumption that the robot is indeed capable of achieving the computed velocity. If that is not the case, then the estimation will rapidly drift from the actual pose. Control integration is well adapted to be used with closed-loop control schemes, such as visual servoing, because they adapt their output to the inaccuracy in the models.

The added perturbations affect the trajectory to reach the goal, which can become heavily deviated. This deviation can be a problem when the robot workspace is cluttered, for example. Moreover, many industrial tasks require being able to follow precisely the chosen trajectory to the goal along with requiring a good accuracy at the goal. In this case trajectory planning and tracking can be used. It allows to not only ensure that the ideal trajectory is closely followed, but also to further increase the robustness to different perturbations. Indeed, in trajectory tracking the desired state is always very close to the current one, which leads to very high robustness of the system. An extensive validation was done on the two CDPRs. It was concluded that the direction of the perturbation affects the robot behavior. That is, having almost the same amplitude of perturbation with a different direction produces different behavior of the system. Nevertheless, when trajectory planning and tracking is used the produced trajectories are always close to the ideal ones. The results are especially noteworthy for the large CDPR CAROCA, where over a distance of 2.3 m the deviation never surpasses 1.5 cm and the mean is 0.71 cm with the standard deviation of 0.34 cm.

Usually, and especially in the work described previously, the cables are not observed and their tensions are not controlled. Thus they can become slack. This can occur due to many reasons, such as the number of cables being greater than the number of degrees of freedom of the moving-platform; drift of moving-platform pose estimation; modeling errors; uncertainties in the system. Cable slack affects the CDPR responsiveness to control output and also its stability. With slack in the system it will need to be transferred between cables as the moving-platform pose changes. During this transfer the moving-platform can become temporarily underactuated, which in turns leads to the stability criterion no longer being ensured. At this moment the moving-platform sways freely, which results in abrupt changes in the image and can cause task failure if the object leaves the image.

Some tension management approaches for visual servoing were proposed and subsequently the tension correction algorithm was implemented. The algorithm is based on comparing the current tension measurement to a threshold and computing a cable velocity correction to reduce the slack. It was shown that TCA successfully detects and removes cable slack without perturbing the main visual servoing controller. Indeed, TCA produces a velocity correction only for the slack cables. These cables are not participating in the control due to the slack and thus the velocity correction does not perturb the resulting moving-platform motion. It was shown that having a TCA allows to greatly improve the ability to track a mobile object. In fact, having all of the cables in tension allowed us to continue tracking the mobile object for several hours, while the visual servo without TCA failed in a matter of minutes.

Perspectives

The perspectives are divided into three groups: (a) short-term goals to extend the current results, building directly on what has been done so far; (b) use cases that are more industry-oriented; (c) work to limit some of the perturbations in the system.

Extension of current results

It was shown that trajectory planning and tracking is an excellent tool to ensure CDPR accuracy all along the trajectory. Similarly, the TCA is a simple and efficient algorithm that keeps the cables in tension and allows the CDPR to execute long-term tasks. Indeed, the TCA deals only with cable slack, however it does not deal with other perturbations in the system and thus the produced trajectory can still be not ideal. Consequently, a combination of TCA and trajectory planning and tracking could be envisioned to have high accuracy, an ideal trajectory and to keep all cables in tension.

A combination of an onboard and one or multiple external cameras could be implemented. The external cameras can be used to measure the moving-platform pose, however as it was shown in Chapter 2, the provided measurement accuracy needs to be high for this to be beneficial. Furthermore, cameras can be used to detect the cable angle and slackness. In the latter case, by detecting the amount of slack on a cable the TCA correction rate can be adjusted to decrease a large slack more rapidly.

Once a good moving-platform pose measurement will be available, it can be of interest to implement some of the more advanced tension management approaches. Indeed, knowing the moving-platform pose a tension distribution algorithm (TDA) could be used to increase the stiffness of the moving-platform.

Use cases

When dealing with real use cases we inevitably encounter constraints and specific requirements coming from the industrial environment. The best way to determine if a CDPR with visual servoing control would be well adapted for a particular task is to implement it on a prototype.

For most of the experiments an AprilTag was used as an object to simplify the computer vision part. In many cases it is not possible to equip the object of interest with an AprilTag or the four-point pattern used for IBVS. Thus, it is of interest to study realistic objects and to define visual servoing control with respect to these objects. Moreover, specific CDPR use cases need to be studied. For example, dealing with occlusion of the field of view during pick-and-place operations. Indeed, it can occur that when the object is picked, it occludes the field of view completely and thus it is not possible to ensure accuracy during the placing of the object. Solutions such as using a pan-tilt camera and changing its angle after picking the object or using multiple cameras can be envisioned.

A sensor fusion can be useful to improve the measurement made by computer vision or to add a missing one. For example, depth sensors can be used to measure the distance along the z axis of the camera. Similarly, an IMU can provide the moving-platform orientation.

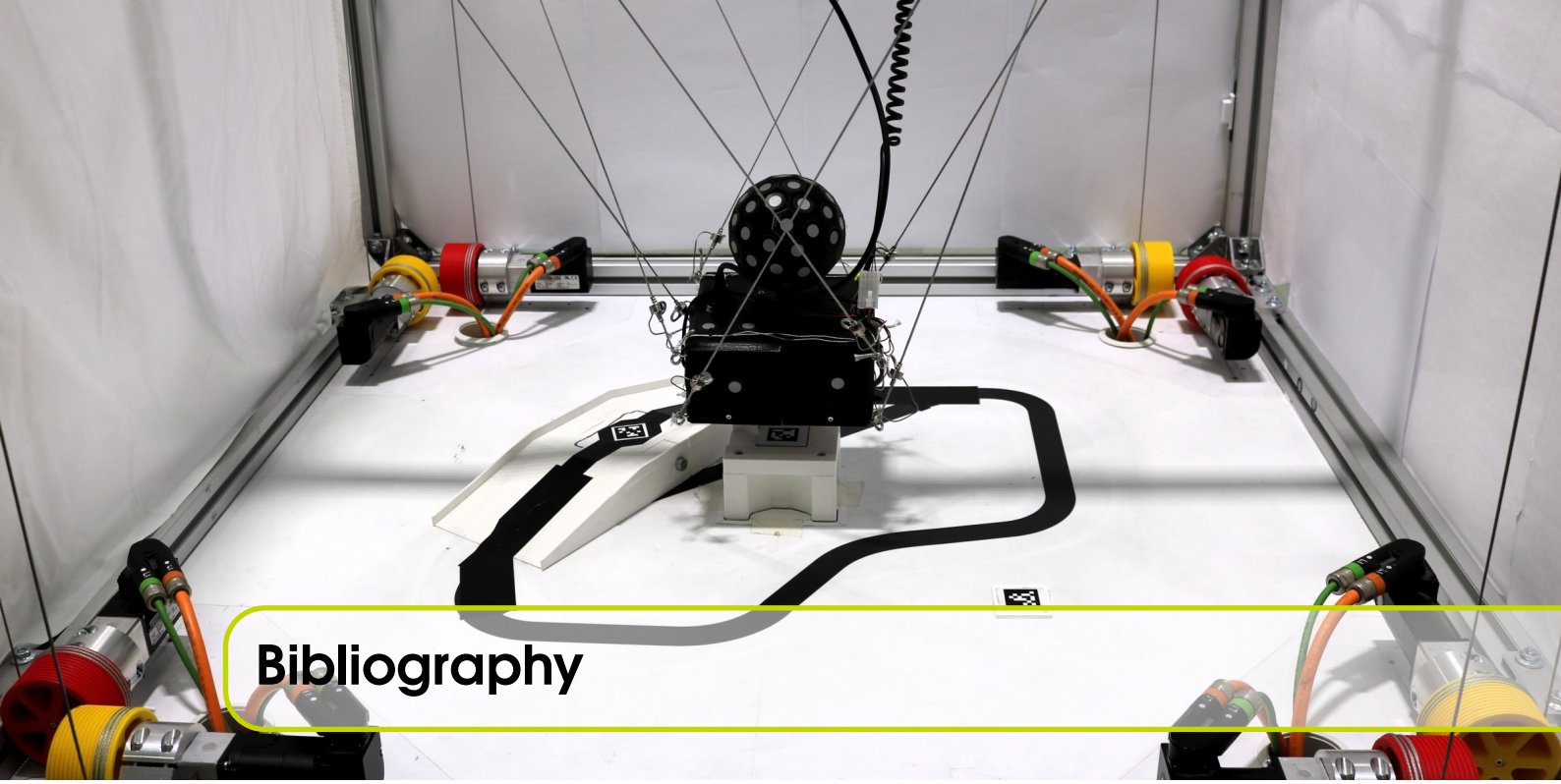
As mentioned above, it is possible that a camera cannot be used during task execution, for example, due to concerns for confidentiality. In this case, other exteroceptive sensors could be used. The possibilities are vast, from infra-red cameras and color sensors, to event cameras, laser trackers and GPS, among many. Thus the generalization of the proposed control approaches to other exteroceptive sensors is indeed of interest.

Controlling the perturbations

A thorough stability study has been performed and it was shown that the different perturbations affect the system in a combined manner. In fact, while having just one perturbation it could be increased to a very high value. On the other hand, when taking into account all perturbations in the system, each of them can only be increased to a limited value before the CSW becomes too small. Thus, by removing some of the perturbations, the robustness to the remaining ones only grows. Therefore, a further improvement would be to develop a control law that allows us to detect and counteract some of these perturbations, for example the modeling errors, instead of increasing control's robustness to them.

Modeling errors can come from not only using the simplified CDPR model, but also from manufacturing. Indeed, the CDPR can simply not correspond to its CAD model. Furthermore, there exist deployable CDPRs that can be used on unknown terrain, making their geometric model badly estimated. Having these model-to-robot differences can seriously affect the system robustness to the remaining perturbations that can occur during task execution. Thus, it would be beneficial to develop a vision-based algorithm to obtain the real cable exit and anchor point coordinates. Indeed, by decreasing these errors, we increase the robustness to other perturbations. This can be crucial in case of the deployable CDPR, where the environment can be highly unpredictable, including weather or smoke. Furthermore, as was shown with large perturbations on cable exit and anchor points, the corners of the workspace become unavailable, while the robot can be required

to work there too. Finally, such an algorithm would also be useful for cases when vision cannot be used during task execution, however can be permitted shortly during CDPR installation.



Bibliography

[ABD92] James Albus, Roger Bostelman, and Nicholas Dagalakis. "The NIST SPIDER, a robot crane". In: *Journal of research of the National Institute of Standards and Technology* 97.3 (1992), p. 373.

[AC15] Ghasem Abbasnejad and Marco Carricato. "Direct Geometrico-static Problem of Underconstrained Cable-Driven Parallel Robots With n Cables". In: *IEEE Transactions on Robotics* 31.2 (2015), pp. 468–478.

[BCC18] Sana Baklouti, Stéphane Caro, and Eric Courteille. "Sensitivity analysis of the elasto-geometrical model of cable-driven parallel robots". In: *Cable-Driven Parallel Robots*. Springer, 2018, pp. 37–49.

[BCG17] Sebastien Briot, Stéphane Caro, and Coralie Germain. "Design procedure for a fast and accurate parallel manipulator". In: *Journal of Mechanisms and Robotics* 9.6 (2017).

[Beg+18] Jeremy Begey, Loic Cuvillon, Maximilien Lesellier, Marc Gouttefarde, and Jacques Gangloff. "Dynamic control of parallel robots driven by flexible cables and actuated by position-controlled winches". In: *IEEE Transactions on Robotics* 35.1 (2018), pp. 286–293.

[Ber15] Alessandro Berti. "Kinematics and statics of cable-driven parallel robots by interval-analysis-based methods". PhD thesis. Université de Nice-Sophia Antipolis, 2015.

[BEU04] Paul Bosscher and Imme Ebert-Uphoff. "A stability measure for underconstrained cable-driven robots". In: *Proceedings - IEEE International Conference on Robotics and Automation*. Vol. 2004. 5. 2004, pp. 4943–4949.

[BG05] Guillaume Barrette and Clément M. Gosselin. "Determination of the dynamic workspace of cable-driven planar parallel mechanisms". In: *Journal of Mechanical Design, Transactions of the ASME* 127.2 (2005), pp. 242–248.

[BK06] Saeed Behzadipour and Amir Khajepour. "Stiffness of cable-based parallel manipulators with application to stability analysis". In: *Journal of Mechanical Design, Transactions of the ASME* 128.1 (2006), pp. 303–310.

[BMC16] Alessandro Berti, Jean Pierre Merlet, and Marco Carricato. "Solving the direct geometrico-static problem of underconstrained cable-driven parallel robots by interval analysis". In: *International Journal of Robotics Research* 35.6 (2016), pp. 723–739.

[BMG97] F. Berry, Philippe Martinet, and Jean Gallice. "Trajectory generation by visual servoing". In: *IEEE International Conference on Intelligent Robots and Systems*. Vol. 2. October. 1997, pp. 1066–1072.

[Bor+09] Per Henrik Borgstrom, Brett L. Jordan, Gaurav S. Sukhatme, Maxim A. Batalin, and William J. Kaiser. "Rapid computation of optimally safe tension distributions for parallel cable-driven robots". In: *IEEE Transactions on Robotics* 25.6 (2009), pp. 1271–1281.

[Bos+07] Paul Bosscher, Robert L. Williams, L. Sebastian Bryson, and Daniel Castro-Lacouture. "Cable-suspended robotic contour crafting system". In: *Automation in Construction*. Vol. 17. 1. 2007, pp. 45–55.

[BREU06] Paul Bosscher, Andrew T. Riechel, and Imme Ebert-Uphoff. "Wrench-feasible workspace generation for cable-driven robots". In: *IEEE Transactions on Robotics* 22.5 (2006), pp. 890–902.

[Bru+06] Tobias Bruckmann, Andreas Pott, Daniel Franitza, and Manfred Hiller. "A modular controller for redundantly actuated tendon-based stewart platforms". In: *1st European Conference on Mechanism Science (EuCoMeS)*. Ed. by Manfred Husty and Hans-Peter Schrocker. Obergurgl, Austria, 2006, pp. 1–12.

[BSW11] Tobias Bruckmann, Christian Sturm, and Lalo Wildan. "Wire Robot Suspension Systems for Wind Tunnels". In: *Wind Tunnels and Experimental Fluid Dynamics Research* (2011).

[BWT05] Paul Bosscher, Robert L. Williams, and Melissa Tummino. "A concept for rapidly-deployable cable robot search and rescue systems". In: *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference - DETC2005*. 2005, pp. 589–598.

[CA13] Marco Carricato and Ghasem Abbasnejad. "Direct Geometrico-Static Analysis of Under-Constrained Cable-Driven Parallel Robots with 4 Cables". In: *Cable-Driven Parallel Robots*. Springer, Berlin, Heidelberg, 2013, pp. 269–285.

[Cab] *CableBOT project*. URL: `http://www.cablebot.eu/en/`.

[Car13] Marco Carricato. "Direct Geometrico-Static Problem of Underconstrained Cable-Driven Parallel Robots with Three Cables". In: *Mechanisms and Machine Science* 5.3 (2013), pp. 269–285.

[CC13] Jean François Collard and Philippe Cardou. "Computing the lowest equilibrium pose of a cable-suspended rigid body". In: *Optimization and Engineering* 14.3 (2013), pp. 457–476.

[CCL15] Ryad Chellal, Loïc Cuvillon, and Edouard Laroche. "A Kinematic Vision-Based Position Control of a 6-DoF Cable-Driven Parallel Robot". In: *Cable-Driven Parallel Robots*. 2015, pp. 213–225.

[CG93] Peter Ian Corke and Malcolm Good. "Controller Design for High-Performance Visual Servoing". In: *IFAC Proceedings Volumes* 26.2 (1993), pp. 629–632.

[CH08] François Chaumette and Seth Hutchinson. "Visual Servoing and visual tracking". In: *Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer, 2008, pp. 563–583.

[Cha04] François Chaumette. "Image Moments: A General and Useful Set of Features for Visual Servoing". In: *IEEE Transactions on Robotics* 20.4 (2004), pp. 713–723.

[Che+01] L. D. Chen, Yiannis Papelis, Ginger Watson, and Dario Solis. "NADS at the University of IOWA: A tool for driving safety research". In: *Proceedings of the 1st Human-Centered Transportation Simulation Conference*. Vol. 2001. Iowa, 2001.

[CM13] Marco Carricato and Jean Pierre Merlet. "Stability analysis of underconstrained cable-driven parallel robots". In: *IEEE Transactions on Robotics* 29.1 (2013), pp. 288–296.

[CRE93] François Chaumette, Patrick Rives, and Bernard Espiau. "Classification and realization of the different vision-based tasks". In: *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*. World Scientific, 1993, pp. 199–228.

[CT] Creaform C-Track. *Technical specifications*. URL: `https://www.creaform3d.com/en/metrology-solutions/3d-applications-software-platforms/dynamic-tracking-vxtrack`.

[Dal+11] Tej Dallej, Marc Gouttefarde, Nicolas Andreff, Micaël Michelin, and Philippe Martinet. "Towards vision-based control of cable-driven parallel robots". In: *International Conference Intelligent Robots and Systems*. San Francisco, United States, 2011, pp. 2855–2860.

[Dal+12] Tej Dallej, Marc Gouttefarde, Nicolas Andreff, Redwan Dahmouche, and Philippe Martinet. "Vision-based modeling and control of large-dimension cable-driven parallel robots". In: *IEEE International Conference on Intelligent Robots and Systems*. Vilamoura, Algarve, Portugal, 2012, pp. 1581–1586.

[Dal+19] Tej Dallej, Marc Gouttefarde, Nicolas Andreff, Pierre Elie Hervé, and Philippe Martinet. "Modeling and vision-based control of large-dimension

cable-driven parallel robots using a multiple-camera setup". In: *Mechatronics* 61 (2019), pp. 20–36.

[DPL07] Lionel Dominjon, Jérôme Perret, and Anatole Lécuyer. "Novel devices and interaction techniques for human-scale haptics". In: *Visual Computer* 23.4 (2007), pp. 257–266.

[Ead14] Ethan Eade. *Lie groups for computer vision*. 2014.

[Fan05] Shiqing Fang. *Design, modeling and motion control of tendon based parallel manipulators*. VDI-Verlag, 2005.

[FCCCL16] Alexis Fortin-Côté, Philippe Cardou, and Alexandre Campeau-Lecours. "Improving cable driven parallel robot accuracy through angular position sensors". In: *IEEE International Conference on Intelligent Robots and Systems*. Daejeon, Korea: IEEE, 2016, pp. 4350–4355.

[FCCG14] Alexis Fortin-Côté, Philippe Cardou, and Clément Gosselin. "An admittance control scheme for haptic interfaces based on cable-driven parallel mechanisms". In: *IEEE International Conference on Robotics and Automation*. 2014, pp. 819–825.

[GA88] Clement Gosselin and Jorge Angeles. "The optimum kinematic design of a planar three-degree-of-freedom parallel manipulator". In: *Journal of Mechanisms, Transmissions, and Automation in Design* 110.1 (1988), pp. 35–41.

[Gag16] Lorenzo Gagliardini. "Discrete Reconfigurations of Cable-Driven Parallel Robots". PhD thesis. Ecole Centrale de Nantes, 2016.

[Gag+16] Lorenzo Gagliardini, Stéphane Caro, Marc Gouttefarde, and Alexis Girin. "Discrete reconfiguration planning for Cable-Driven Parallel Robots". In: *Mechanism and Machine Theory* 100 (2016), pp. 313–337.

[GDM11] Marc Gouttefarde, David Daney, and Jean-pierre Merlet. "Interval-Analysis-Based Determination of the Wrench-Feasible Workspace of Parallel Cable-Driven Robots". In: *IEEE Transactions on Robotics* 27.1 (2011), pp. 1–13.

[GGC18] Lorenzo Gagliardini, Marc Gouttefarde, and Stephane Caro. "Determination of a dynamic feasible workspace for cable-driven parallel robots". In: *Advances in Robot Kinematics 2016*. Vol. 4. Springer, 2018, pp. 361–370.

[GJC01] Jason J Gorman, Kathryn W Jablokow, and David J Cannon. "The cable array robot: Theory and experiment". In: *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation*. Vol. 3. IEEE. 2001, pp. 2804–2810.

[Gou+12] Marc Gouttefarde, Jean François Collard, Nicolas Riehl, and Cédric Baradat. "Simplified static analysis of large-dimension parallel cable-driven robots". In: *Proceedings - IEEE International Conference on Robotics and Automation*. 2012, pp. 2299–2305.

[Gou+15] Marc Gouttefarde, Johann Lamaury, Christopher Reichert, and Tobias Bruckmann. "A Versatile Tension Distribution Algorithm for n-DOF Parallel

Robots Driven by n + 2 Cables". In: *IEEE Transactions on Robotics* 31.6 (2015), pp. 1444–1457.

[Gua+13] François Guay, Philippe Cardou, Ana Lucia, Cruz Ruiz, and Stéphane Caro. "Measuring How Well a Structure Supports Varying External Wrenches". In: *The Second Conference on Mechanisms, Transmissions and Applications (MeTrApp 2013)*. 2013.

[Han] *HangPrinter printing Babel Tower*. URL: `https://vitana.se/opr3d/tbear/2017.html#hangprinter_project_40`.

[HK11] Mahir Hassan and Amir Khajepour. "Analysis of bounded cable tensions in cable-actuated parallel manipulators". In: *IEEE Transactions on Robotics* 27.5 (2011), pp. 891–900.

[Ho+14] Wei Yang Ho, Werner Kraus, Alexander Mangold, and Andreas Pott. "Haptic Interaction with a Cable-Driven Parallel Robot Using Admittance Control". In: *Cable-Driven Parallel Robots*. Springer, Cham, 2014, pp. 201–212.

[Hu+14] Yongpan Hu, Limin Tao, Jun Jia, and Wei Lv. "Control and simulation of cable-driven parallel robots in offshore cargo handling". In: *Proceeding of the 11th world congress on intelligent control and automation.* IEEE, 2014, pp. 2451–2455.

[Ids] *IDS homepage*. URL: `https://en.ids-imaging.com/home.html`.

[Irv92] H. Max Irvine. *Cable Structures*. New York: Dover Publications, 1992.

[Iza+13] Jean-Baptiste Izard, Marc Gouttefarde, Micaël Michelin, Olivier Tempier, and Cedric Baradat. "A Reconfigurable Robot for Cable-Driven Parallel Robotic Research and Industrial Scenario Proofing". In: *Cable-Driven Parallel Robots*. Springer, Berlin, Heidelberg, 2013, pp. 135–148.

[Iza+17] Jean-Baptiste Izard, Alexandre Dubor, Pierre-Elie Hervé, Edouard Cabay, David Culla, Mariola Rodriguez, and Mikel Barrado. "Large-scale 3D printing with cable-driven parallel robots". In: *Construction Robotics* 1.1-4 (2017), pp. 69–76.

[Joh70] C R Johnson. "Positive definite matrices". In: *The American Mathematical Monthly* 77.3 (1970), pp. 259–264.

[Kaw+97] Sadao Kawamura, Won Choe, Satoshi Tanaka, and Hitoshi Kino. "Development of an ultrahigh speed robot FALCON using parallel wire drive systems". In: *Journal of the Robotics Society of Japan* 15.1 (1997), pp. 82–89.

[KD04] Wisama Khalil and Etienne Dombre. *Modeling, identification and control of robots*. Butterworth-Heinemann, 2004.

[Kha02] Hassan K. Khalil. *Nonlinear Systems*. 2002.

[KI93] Sadao Kawamura and Ken Ito. "A new type of master robot for teleoperation using a radial wire drive system". In: *IEEE International Conference on Intelligent Robots and Systems*. 1993, pp. 55–60.

[KKC04] Ville Kyrki, Danica Kragic, and Henrik I. Christensen. "New shortest-path approaches to visual servoing". In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* 1 (2004), pp. 349–354.

[Kow] *KOWA lens specification*. URL: `https://www.rmaelectronics.com/kowa-lm5ncl/`.

[Koz14] V. Kozlov. "A graphical user interface for the design of cable-driven parallel robots". Master. École Centrale de Nantes, 2014.

[Kra+14] Werner Kraus, Valentin Schmidt, Puneeth Rajendra, and Andreas Pott. "System identification and cable force control for a cable-driven parallel robot with industrial servo drives". In: *Proceedings - IEEE International Conference on Robotics and Automation*. Hong Kong, China, 2014, pp. 5921–5926.

[KZW06] Kris Kozak, Qian Zhou, and Jinsong Wang. "Static analysis of cable-driven manipulators with non-negligible cable mass". In: *IEEE Transactions on Robotics* 22.3 (2006), pp. 425–433.

[Lam14] Johann Lamaury. "Contribution à la commande des robots parallèles à câbles à redondance d'actionnement". PhD thesis. Université Montpellier II, 2014.

[LG13] Johann Lamaury and Marc Gouttefarde. "Control of a Large Redundantly Actuated Cable-Suspended Parallel Robot". In: *IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 4659–4664.

[LLR02] Pascal Lafourcade, Michel Llibre, and Claude Reboulet. "Design of a Parallel Wire-Driven Manipulator for Wind Tunnels". In: *Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators* (2002), pp. 187–194.

[Lud] Torbjørn Ludvigsen. *HangPrinter*. URL: `https://hangprinter.org/`.

[MADS15] Jean Pierre Merlet and Julien Alexandre-Dit-Sandretto. "The forward kinematics of cable-driven parallel robots with sagging cables". In: *Cable-Driven Parallel Robots*. 2015, pp. 3–15.

[MC02] Youcef Mezouar and François Chaumette. "Path planning for robust image-based control". In: *IEEE Transactions on Robotics and Automation* (2002), pp. 534–549.

[MC07] Nicolas Mansard and François Chaumette. "Task sequencing for sensor-based control". In: *IEEE Transactions on Robotics* 23.1 (2007), pp. 60–72.

[MCB99] Ezio Malis, François Chaumette, and Sylvie Boudet. "2 1/2 D Visual Servoing". In: *IEEE Tran on Robotics and Automation* 15.2 (1999), pp. 238–250.

[MD10] Jean Pierre Merlet and David Daney. "A portable, modular parallel wire crane for rescue operations". In: *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE. 2010, pp. 2834–2839.

[Mer06] Jean Pierre Merlet. *Parallel Robots*. Vol. 53. 9. Springer, 2006.

[Mer17] Jean Pierre Merlet. "Simulation of discrete-time controlled cable-driven parallel robots on a trajectory". In: *IEEE Transactions on Robotics* 33.3 (2017), pp. 675–688.

[Mie+16] Philipp Miermeister, Maria Lächele, Rainer Boss, Carlo Masone, Christian Schenk, Joachim Tesch, Michael Kerger, Harald Teufel, Andreas Pott, and Heinrich H Bülthoff. "The cablerobot simulator large scale motion platform based on cable robot technology". In: *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE. 2016, pp. 3024–3029.

[Mik+08] Lars Mikelsons, Tobias Bruckmann, Manfred Hiller, and Dieter Schramm. "A real-time capable force calculation algorithm for redundant tendon-based parallel manipulators". In: *Proceedings - IEEE International Conference on Robotics and Automation*. Pasadena, CA, USA, 2008, pp. 3869–3874.

[MSC05] Éric Marchand, Fabien Spindler, and François Chaumette. "ViSP for visual servoing: A generic software platform with a wide class of robot control skills". In: *IEEE Robotics and Automation Magazine* 12.4 (2005), pp. 40–52.

[MT19] Media-Tech. *AUTOPIX MT4018 User's Guide*. 2019. URL: `https://www.manualslib.com/manual/1220424/Media-Tech-Autopix-Mt4018.html`.

[NG14] Dinh Quan Nguyen and Marc Gouttefarde. "Study of Reconfigurable Suspended Cable-Driven Parallel Robots for Airplane Maintenance". In: *IEEE International Conference on Intelligent Robots and Systems*. Chicago, IL, United States, 2014, pp. 1682–1689.

[NL11] Rendong Nan and Di Li. "The five-hundred-meter aperture spherical radio telescope (FAST) project". In: *International Journal of Modern Physics D* 20.06 (2011), pp. 989–1024.

[OA04] So Ryeok Oh and Sunil Kumar Agrawal. "Nonlinear sliding mode control and feasible workspace analysis for a cable suspended robot with input constraints and disturbances". In: *Proceedings of the American Control Conference* 5.4 (2004), pp. 4631–4636.

[Ols11] Edwin Olson. "AprilTag: A robust and flexible visual fiducial system". In: *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3400–3407.

[Orb] Orbbec. *Astra camera*. URL: `https://orbbec3d.com/product-astra-pro/`.

[PCM05] Jonghoon Park, Wan Kyun Chung, and Wonkyu Moon. "Wire-suspended dynamical system: Stability analysis by tension-closure". In: *IEEE Transactions on Robotics* 21.3 (2005), pp. 298–308.

[Per+10] Simon Perreault, Philippe Cardou, Clément M Gosselin, and Martin J-D Otis. "Geometric determination of the interference-free constant-orientation workspace of parallel cable-driven mechanisms". In: *Journal of Mechanisms and Robotics* 2.3 (2010).

[Pic18] Etienne Picard. "Contributions à la modélisation, étalonnage et commande robuste de robots parallèles à câbles". PhD thesis. Universite Bretagne Loire, 2018.

[Pic+18a] Etienne Picard, Stéphane Caro, Fabien Claveau, and Franck Plestan. "Pulleys and Force Sensors Influence on Payload Estimation of Cable-Driven Parallel

Robots". In: *IEEE International Conference on Intelligent Robots and Systems* (2018), pp. 1429–1436.

[Pic+18b] Etienne Picard, Stéphane Caro, Franck Plestan, and Fabien Claveau. "Control Solution for a Cable Driven Parallel Robot with highly variable payload". In: *ASME 2018 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*. Quebec, 2018, pp. 1429–1436.

[Pol] *Polargraph*. URL: `https://trmm.net/Polargraph`.

[Pot12] Andreas Pott. "Influence of pulley kinematics on cable-driven parallel robots". In: *Latest Advances in Robot Kinematics* (2012), pp. 197–204.

[Pot14] Andreas Pott. "An Improved Force Distribution Algorithm for Over-Constrained Cable-Driven Parallel Robots". In: *Computational Kinematics*. Dordrecht: Springer, 2014, pp. 139–146.

[Pot18] Andreas Pott. *Cable-Driven Parallel Robots: Theory and Application*. Springer, 2018.

[Pot+19] Andreas Pott, Philipp Tempel, Alexander Wilhelm Verl, and Frederik Wulle. "Design, Implementation and Long-Term Running Experiences of the Cable-Driven Parallel Robot CaRo Printer". In: *Cable-Driven Parallel Robots*. Springer, Cham, 2019, pp. 379–390.

[PVF20] Aidan Phillips, Ashwin Vinoo, and Naomi T. Fitter. "May i Draw Your Attention? Initial Lessons from the Large-Scale Generative Mark Maker". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 690–697.

[Ras+18] Tahir Rasheed, Philip Long, David Marquez-Gamez, and Stéphane Caro. "Tension distribution algorithm for planar mobile cable-driven parallel robots". In: *Cable-Driven Parallel Robots*. Springer, Cham, 2018, pp. 268–279.

[Ras19] Tahir Rasheed. "Collaborative Mobile Cable-Driven Parallel Robots". PhD thesis. École centrale de Nantes, 2019.

[RCM14] Rémy Ramadour, François Chaumette, and Jean Pierre Merlet. "Grasping objects with a cable-driven parallel robot designed for transfer operation by visual servoing". In: *Proceedings - IEEE International Conference on Robotics and Automation*. Hong Kong, 2014, pp. 4463–4468.

[RGL98] Rodney G. Roberts, Todd Graham, and Thomas Lippitt. "On the inverse kinematics, statics, and fault tolerance of cable-suspended robots". In: *Journal of Robotic Systems* 15.10 (1998), pp. 581–597.

[Rie11] Nicolas Riehl. "Modélisation et conception de robots parallèles à câbles de grande dimension". PhD. Université Montpellier II, 2011.

[RSF13] Eric Rohmer, Surya P N Singh, and Marc Freese. "V-REP: A versatile and scalable robot simulation framework". In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, pp. 1321–1326.

[Rui+15] Ana Lucia Cruz Ruiz, Stéphane Caro, Philippe Cardou, and François Guay. "ARACHNIS : Analysis of Robots Actuated by Cables with Handy and

Neat Interface Software". In: *Cable-Driven Parallel Robots*. Springer, 2015, pp. 293–305.

[Sch+17] Valentin Schmidt, Werner Kraus, Christoph Martin, Xuejun Jin, and Andreas Pott. "Black-box accuracy compensation for a cable-driven parallel robot". In: *International Conference on Control, Automation and Systems*. 2017, pp. 428–431.

[Sch17] Valentin Lorenz Schmidt. "Modeling Techniques and Reliable Real-Time Implementation of Kinematics for Cable-Driven Parallel Robots using Polymer Fiber Cables." PhD thesis. Stuttgart: Fraunhofer Verlag, 2017.

[Scr] *Scribit Wall Plotter*. URL: `https://scribit.design/products/scribit`.

[Sic+10] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics. Modelling, planning, control*. Ed. by Michael Grimble and Michael Johnson. Springer, 2010.

[SK06] Ethan Stump and Vijay Kumar. "Workspaces of cable-actuated parallel manipulators". In: *Journal of Mechanical Design* 128.1 (2006), pp. 159–167.

[Sky] *SKYCAM*. URL: `http://skycam.tv/`.

[SR19] Dragoljub Surdilovic and Jelena Radojicic. "Practical stability of under-constrained cable-suspended parallel robots". In: *Cable-Driven Parallel Robots. CableCon 2019. Mechanisms and Machine Science*. Springer, Cham, 2019, pp. 85–98.

[Tec] Stellar Tech. *VLU850 tension sensors*. URL: `https://www.stellartech.com/products/techsheets-load/vlu850.pdf`.

[Tra] Tractel. *Dynasafe force sensors*. URL: `https://www.tractel.com/fr/product/dynasafe-electroniques/6916`.

[VB06] Martin P Vlasblom and Rigo LM Bosman. "Predicting the creep lifetime of HMPE mooring rope applications". In: *OCEANS 2006*. IEEE. 2006, pp. 1–10.

[Ver] IRT Jules Verne. *Video of Project DECAUV*. URL: `https://www.youtube.com/watch?v=A3L9MK8T30U`.

[Ver04] Richard Verhoeven. "Analysis of the Workspace of Tendon-Based". PhD thesis. Duisburg Gerhard Mercator University, 2004.

[ViS19] ViSP. *Tutorial: How to boost your visual servo control law*. 2019. URL: `https://visp-doc.inria.fr/doxygen/visp-daily/tutorial-boost-vs.html`.

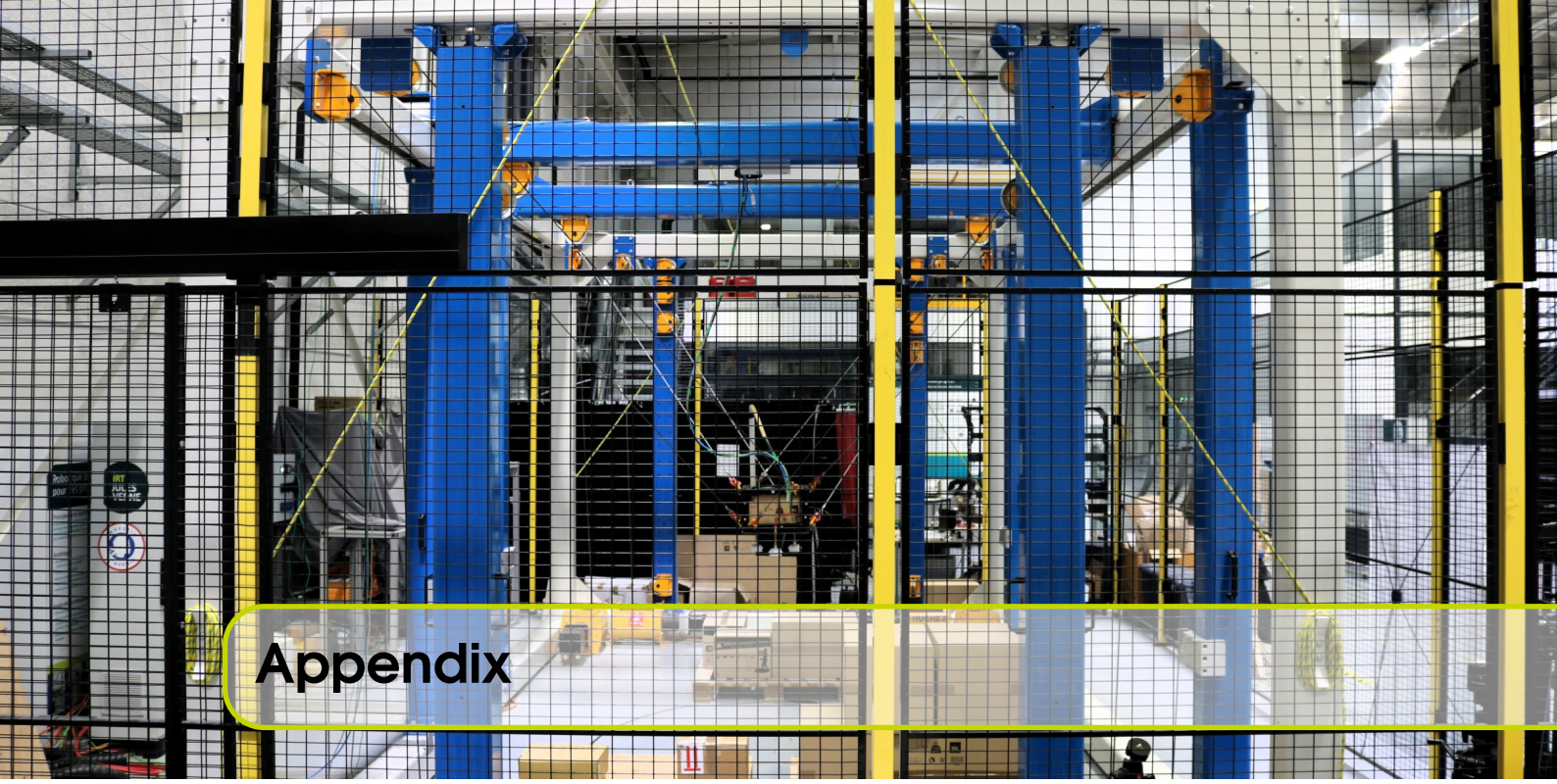
[Viv] HTC Vive. *tracker*. URL: `https://www.vive.com/eu/accessory/vive-tracker/`.

[VR19] V-REP. *Tutorial: Designing Dynamic Simulations*. 2019. URL: `http://www.coppeliarobotics.com/helpFiles/en/designingDynamicSimulations.htm`.

[Web16] Xavier Weber. "Commande modale de robots parallèles à câbles flexibles". PhD thesis. Université de Strasbourg, 2016.

[Yan+10] Xiao Yangwen, Lin Qi, Zheng Yaqing, and Liang Bin. "Model aerodynamic tests with a wire-driven parallel suspension system in low-speed wind tunnel". In: *Chinese Journal of Aeronautics* 23.4 (2010), pp. 393–400.

[YY09] Yang Yi and Zhang Yuru. "A new cable-driven haptic device for integrating kinesthetic and cutaneous display". In: *ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots (ReMAR)*. 2009, pp. 386–391.



Appendix

Contents

A.1 CDPRs at IRT Jules Verne

A.1.1 ACROBOT

ACROBOT, shown in Fig. A.1, is a small CDPR available at IRT Jules Verne, Nantes. It was developed as a demonstration prototype that is small enough to be transported to exhibitions.

Geometry

ACROBOT's frame is a cube of 1.2 m×1.2 m×1.2 m. It is assembled in a suspended configuration. It has six DoF that are actuated by eight Dyneema SK78 cables of 2 mm diameter. The Young Modulus of these cables is 111 GPa [VB06]. The cables are wound on winches with radius $r_w = 0.04\,\text{m}$.

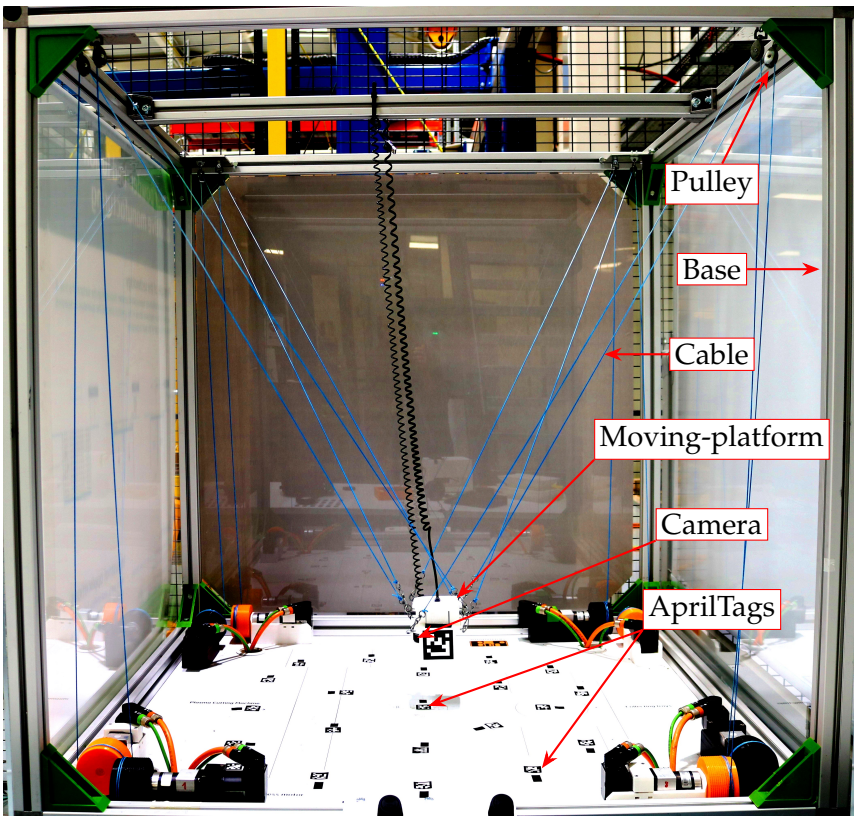


Figure A.1: ACROBOT: a CDPR prototype located at IRT Jules Verne, Nantes

The origin of the base frame $\mathcal{F}_b$ is assumed to be at the center of the $\mathbf{xy}$ plane, on the floor of the workspace. The coordinates of cable exit points in base frame $\mathcal{F}_b$ and anchor points in moving-platform frame $\mathcal{F}_p$, are presented in Table A.1. The first set of cable anchor points corresponds to the small 11 cm×11 cm×7 cm and 1.5 kg moving-platform visible in Fig. 1.9a. The second corresponds to a larger 18 cm×17 cm×7 cm and 3.5 kg moving-platform that houses not only a camera, but also a set of eight tension sensors and their transmitters (see Fig. A.2). For simplicity, cable anchor points are called B1 ... B8 for both moving-platforms, because they cannot be used simultaneously.

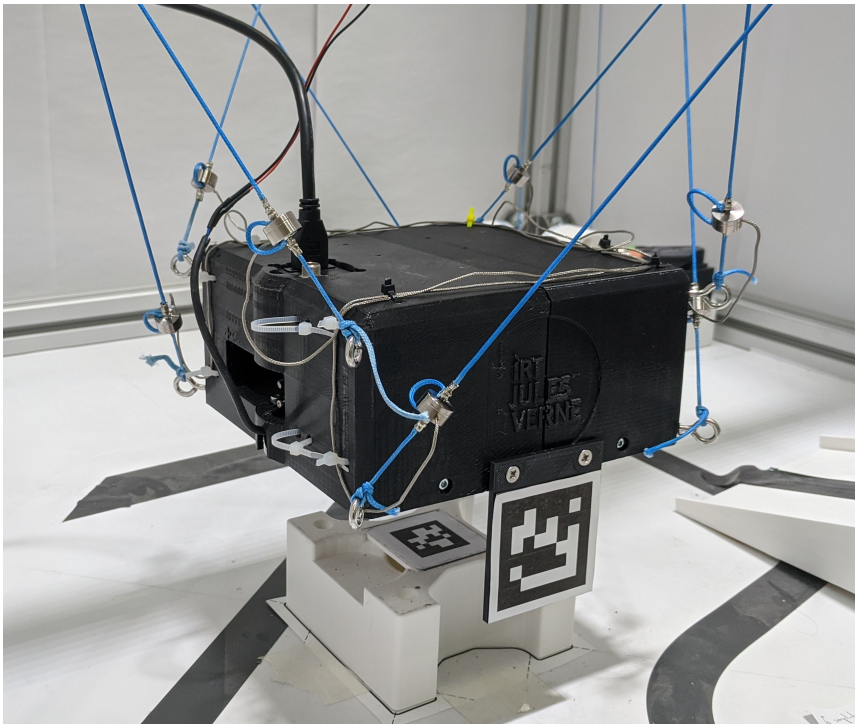


Figure A.2: Large moving-platform for ACROBOT housing a camera and eight tension sensors

Table A.1: Coordinates of ACROBOT cable exit and anchor points

Cable exit points expressed in $\mathcal{F}_b$, m			
A1	$[-0.47; -0.525; 1.165]^\top$	A5	$[0.47; 0.53; 1.165]^\top$
A2	$[-0.52; -0.465; 1.165]^\top$	A6	$[0.52; 0.475; 1.165]^\top$
A3	$[-0.52; -0.475; 1.165]^\top$	A7	$[0.525; -0.47; 1.165]^\top$
A4	$[-0.47; 0.53; 1.165]^\top$	A8	$[0.47; -0.520; 1.165]^\top$
Cable anchor points expressed in $\mathcal{F}_p$ for the small moving-platform, m			
B1	$[0.055; -0.055; 0.07]^\top$	B5	$[-0.055; 0.055; 0.07]^\top$
B2	$[-0.055; 0.055; 0.025]^\top$	B6	$[0.055; -0.055; 0.025]^\top$
B3	$[-0.055; -0.055; 0.07]^\top$	B7	$[0.055; 0.055; 0.07]^\top$
B4	$[0.055; 0.055; 0.025]^\top$	B8	$[-0.055; -0.055; 0.025]^\top$
Cable anchor points expressed in $\mathcal{F}_p$ for the large moving-platform, m			
B1	$[0.0895; -0.087; 0.072]^\top$	B5	$[-0.0895; 0.087; 0.072]^\top$
B2	$[-0.0895; 0.087; 0.0065]^\top$	B6	$[0.0895; -0.087; 0.0065]^\top$
B3	$[-0.0895; -0.087; 0.072]^\top$	B7	$[0.0895; 0.087; 0.072]^\top$
B4	$[0.0895; 0.087; 0.0065]^\top$	B8	$[-0.0895; -0.087; 0.0065]^\top$

Actuation

ACROBOT is equipped with four synchronous motors with a holding brake and four synchronous motors without a holding brake. The maximum speed of the motors amounts

to 3000 RPM and their gearbox reduction is $R_g = 10$. All motors are fixed to the ground. Winches are mounted directly on motor axis.

Sensors

The small moving-platform is only equipped with one camera. Depending on the experiment, it is either a Media-Tech AUTOPIX MT4018 [MT19] or UI-3250CP-C-HQ Rev.2 by IDS [Ids]. The large moving-platform is equipped with the IDS camera and also with eight VLU850 tension sensors by Stellar Tech [Tec] used for cable tension measurement. Finally, another camera, namely Astra by Orbbec [Orb], is fixed statically in the workspace to observe the moving-platform. The HTC Vive tracking [Viv] system is used to measure the moving-platform pose during experiments, to serve as ground-truth when Creaform C-Track was not available.

The choice of RGB cameras without depth-sensing is due to the size of ACROBOT. Most of the time the moving-platform would be too close to the object (or ground) to use a depth sensing camera on the moving-platform.

Media-Tech AUTOPIX MT4018 For the initial studies, especially for the proof of concept, the simplest webcam by Media-Tech was used, because it was readily available at IRT JV. It is shown in Fig. A.3 and its specification is detailed in Table A.2. It should be noted that the auto focus feature is not controllable, which, as it was found in the experiments, led to quite a bit of noise in the system due to rather frequent zooming in and out.

Table A.2: Media-Tech AUTOPIX MT4018 specifications, taken from [MT19]

Image Sensor	1280x1024 pixels, CMOS
Lens Specification	F2.8, f=4.02 mm, auto-focus lens
White Balance	Auto
Exposure	Auto
Frame Rate (up to)	1280×1024 @ 8~10fps 640×480 @ 30fps
Focus Range	Auto focus, 7 cm to infinity
Depth of Field	50 cm to infinity
PC Interface	USB2.0
Power	From USB port

IDS UI-3250CP-C-HQ Rev.2 Considering the drawbacks of the webcam, it was decided to use a more advanced camera. The necessary improvements were: (i) better actual frame-rate; (ii) better image quality; (iii) no auto-focus (or at least one that is controllable). Thus UI-3250CP-C-HQ Rev.2 by IDS was chosen, it is shown in Fig. A.4a

Figure A.3: Media-Tech AUTOPIX MT4018 web-camera

and the specifications are presented in Table A.3. For simplicity, it is called the IDS camera in the text.

Table A.3: IDS UI-3250CP-C-HQ Rev.2 specifications, taken from [Ids]

Image Sensor	1600 × 1200 pixels, CMOS
White Balance	Modifiable
Exposure	Modifiable
Frame Rate	60fps
Shutter	Global and Rolling
PC Interface	USB3.0
Power	From USB port

Industrial cameras, such as the IDS, can be paired with a wide range of lenses. Here, we use KOWA LM5NCL lens that is shown in Fig. A.4b. Its specifications are presented in Table A.4.

Table A.4: KOWA LM5NCL specifications, taken from [Kow]

Focal Length	4.5 mm
Iris Range	F1.4–F16, manual control
Focusing Range	0.2 m–∞, manual control
Angle of View	H = 79°; V = 59.4°; Diag. = 98.2°
Mount	C-mount

Finally, to improve the light conditions for the visual servoing, a ring light is mounted on the lens (Fig. A.4c).

Orbbec Astra Another camera was necessary for the eye-to-hand configuration and to record videos of the robot during task execution. The Astra camera by Orbbec was used

(a) (b) (c)

Figure A.4: (a) IDS camera, (b) KOWA lens and (c) red light ring

(Fig. A.5). Its specifications are shown in Table A.5. It is a RGB-D camera, i.e. the fourth channel contains depth information, however this has not been used in the scope of this thesis.

Table A.5: Orbbec Astra specifications, taken from [Orb]

Image Resolution	640 × 480 pixels
Frame Rate	30fps
Angle of View	H = 60°; V = 49.5°; Diag. = 73°
Range	0.6 m–8 m
PC Interface	USB2.0
Power	From USB port

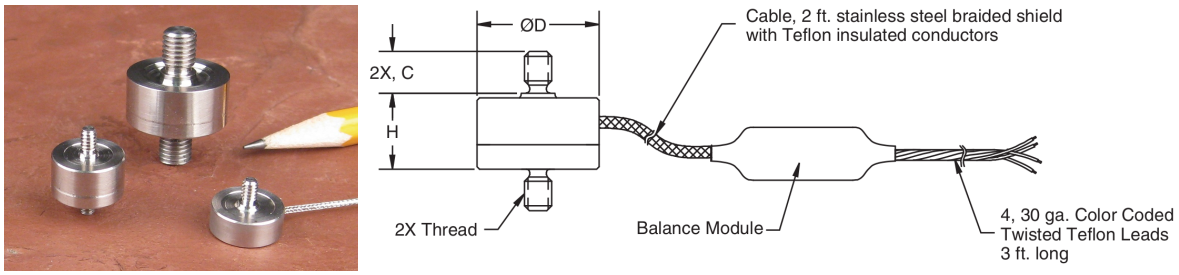
Figure A.5: Orbbec Astra camera

Stellar Tech VLU850 The tension sensors are shown in Fig. A.6. These are very small tension sensors with 50 Lb load capacity. Most important characteristics are detailed in Table A.6. They are attached to the cables close to their anchor points to the moving-platform.

HTC Vive tracking system During the evaluation of different moving-platform pose estimation methods, HTC Vive tracking system, shown in Fig. A.7a, was used as an online measurement tool for the sake of comparison. HTC Vive is a virtual reality headset developed by HTC. While it is mainly advertised as an entertainment product, it is also

Table A.6: Stellar Tech VLU850 tension sensor specifications, taken from [Tec]

Type	Tension and Compression, calibrated for tension
Load range	0–50 Lb, recalibrated to 0–20 Lb for higher accuracy
Max. Overload	75 Lb
Accuracy	0.5% of full scale output
Repeatability	0.01% of full scale output
Size (see Fig. A.6)	⌀ D = 12.7 mm; H = 8 mm, C = 4 mm
Thread	4–40 UNC



(a) **(b)**

Figure A.6: Stellar Tech VLU850 tension sensor: **(a)** image and **(b)** technical drawing

widely used for academic purposes. Indeed, it is a low-cost system that is easy to set up and that provides a decent measurement accuracy, especially for static or slowly moving objects. Of course, it does not provide the same accuracy as a laser tracker, but often a millimetric accuracy is good enough.

A standard Vive set includes the headset, two controllers and two base stations shown in Fig. A.7a. However, for robotics the tracker shown in Fig. A.7b is more convenient. Indeed, it can easily be attached to the robot end-effector or another object of interest.

A tracker is mounted on the moving-platform and two base stations are positioned to cover the whole workspace as shown in Fig. A.7c. It is possible to retrieve the moving-platform pose with respect to the base frame, once the pose of the Vive tracker in the base frame is known. The Vive tracker pose in the base frame is retrieved as follows:

- Setup the tracking system, for example, as shown in Fig. A.7c and launch the internal calibration of the tracker. At this point the tracker pose in the Vive reference frame $\mathcal{F}_v$ becomes available. It can be expressed as ${}^{v}\mathbf{T}_{tr}$;
- Retrieve the pose of the tracker in the moving-platform frame $\mathcal{F}_p$ from the CAD model and express it as ${}^{p}\mathbf{T}_{tr}$;
- Measure the moving-platform pose in the base frame $\mathcal{F}_b$ with C-Track and express it as ${}^{b}\mathbf{T}_{p}$;
- Then the origin of $\mathcal{F}_v$ can be expressed in $\mathcal{F}_b$ as: ${}^{b}\mathbf{T}_{v} = {}^{b}\mathbf{T}_{p}\,{}^{p}\mathbf{T}_{tr}\,{}^{v}\mathbf{T}_{tr}^{-1}$

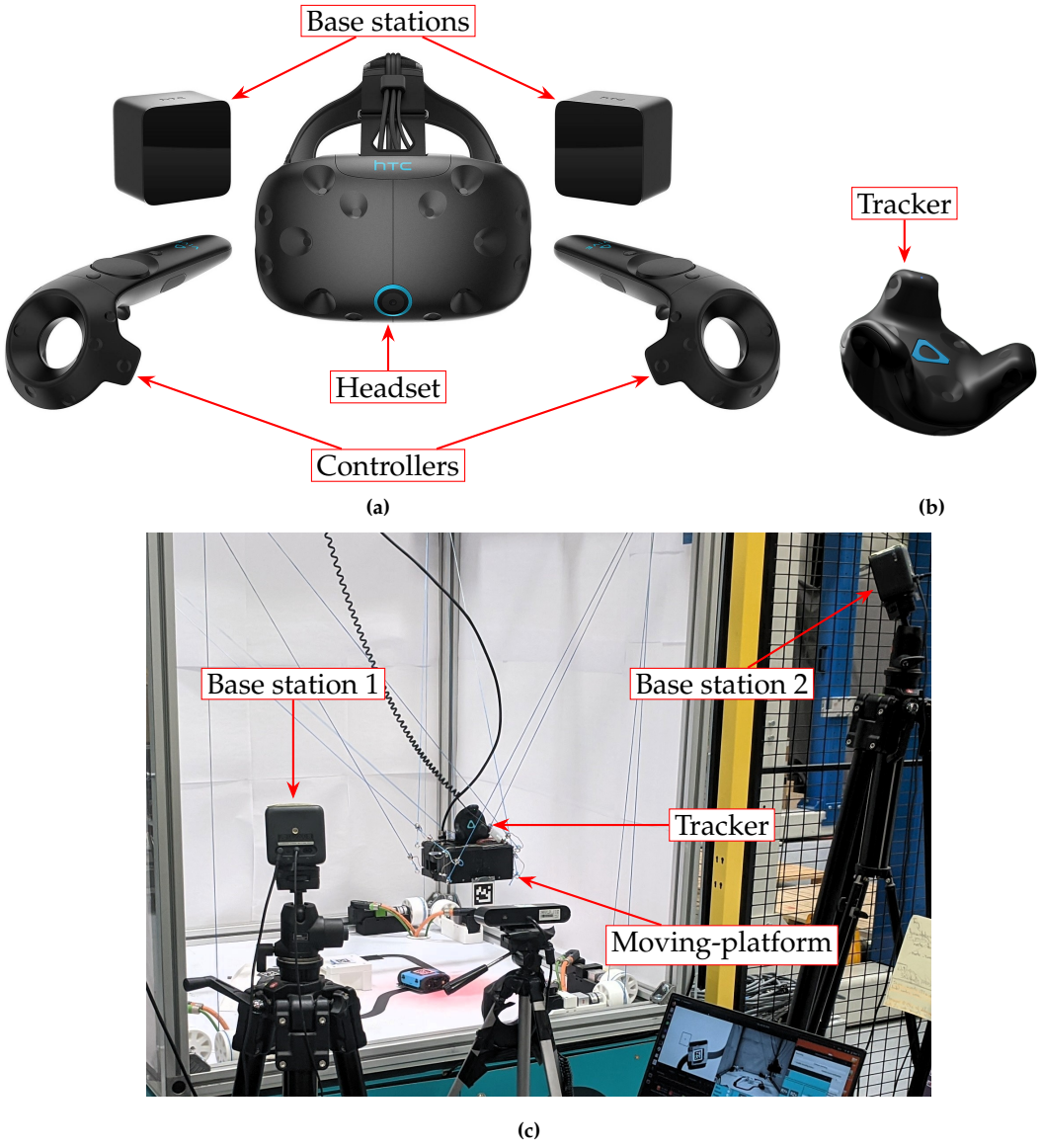


Figure A.7: HTC Vive tracking system: **(a)** standard Vive kit; **(b)** tracker; **(c)** tracker mounted on the moving-platform and base stations located in the close vicinity

- Finally, at each time instant t the moving-platform pose as measured by the Vive tracker becomes: ${}^{b}\mathbf{T}_{p}(t) = {}^{b}\mathbf{T}_{v}{}^{v}\mathbf{T}_{tr}(t){}^{p}\mathbf{T}_{tr}^{-1}$

A.1.2 CAROCA

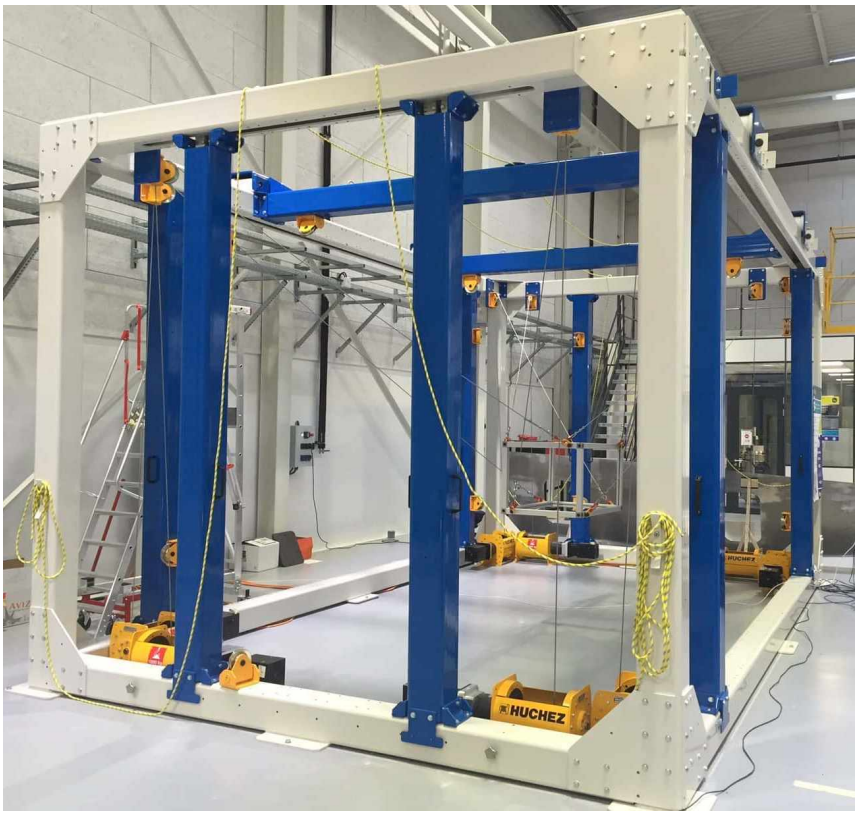
CAROCA is a large reconfigurable CDPR located at IRT Jules Verne, Nantes. It was developed to research the possibility of CDPR industrialization for tasks such as photogrammetry, sandblasting and painting of large and complex parts, pick and place operations [Pic18]. The reconfiguration strategies of CAROCA were studied in [Gag16].

Geometry

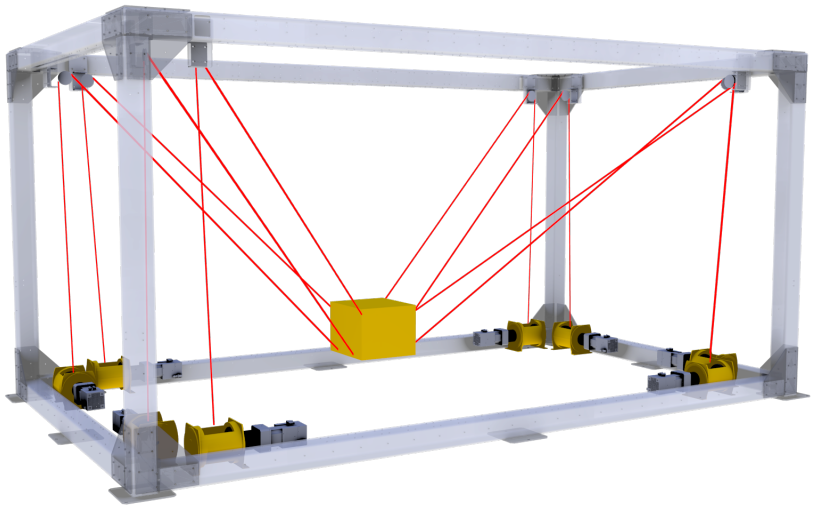
CAROCA, shown in Fig. A.8, has the following dimensions: 7 m×4 m×3 m. It has six DoF and is connected to the base by eight steel cables of 6.0 mm diameter. The Young

Modulus of these cables has been experimentally evaluated at 102.2 GPA [BCC18]. The cables are wound on winches with radius $r_w = 0.06\,\text{m}$.

Unlike ACROBOT, here the pulleys at cable exit points are of non-negligible size, i.e. pulley radius is 150 mm. Furthermore, steel cables are heavy and elastic, thus a



(a)



(b)

Figure A.8: A large six DoF CDPR named CAROCA: **(a)** image and **(b)** CAD model [Pic18]

sagging elastic cable model should be used for this robot. CAROCA can be assembled in a suspended or a fully constrained configuration. Moreover, due to its reconfigurable frame, it is possible to change pulley location by moving the blue columns in the frame and the pulleys themselves on these blue columns (see Fig. A.8a). While in a suspended configuration, CAROCA is capable of lifting up to one ton [Pic18].

The moving-platform of CAROCA is 42 cm×32 cm×23 cm and its mass is 150 kg. It is equipped with one onboard camera and eight tensions sensors for cable tension measurement. Cable exit and anchor point coordinates are given in Table A.7.

Table A.7: Coordinates of CAROCA cable exit and anchor points

Cable exit points expressed in $\mathcal{F}_b$, m			
A1	$[1.659; -2.850; 3.221]^\top$	A5	$[-1.659; 2.850; 3.221]^\top$
A2	$[1.350; -3.159; 3.221]^\top$	A6	$[-1.350; 3.159; 3.221]^\top$
A3	$[-1.350; -3.159; 3.221]^\top$	A7	$[1.350; 3.159; 3.221]^\top$
A4	$[-1.659; -2.850; 3.221]^\top$	A8	$[1.659; 2.850; 3.221]^\top$
Cable anchor points expressed in $\mathcal{F}_p$, m			
B1	$[0.21; 0.13; 0.23]^\top$	B5	$[-0.21; -0.13; 0.23]^\top$
B2	$[-0.14; -0.16; 0.02]^\top$	B6	$[0.14; 0.16; 0.02]^\top$
B3	$[0.14; -0.16; 0.23]^\top$	B7	$[-0.14; 0.16; 0.23]^\top$
B4	$[-0.21; 0.13; 0.02]^\top$	B8	$[0.21; -0.13; 0.02]^\top$

Actuation

The cables of CAROCA are actuated by eight synchronous motors of nominal speed equal to 2200 RPM and nominal torques equal to 15.34 Nm. Each motor has a gearbox with reduction ratio $R_g = 40$.

Sensors

The moving-platform of CAROCA can be equipped with the IDS camera described in Appendix A.1.1.

The moving-platform is equipped with eight Tractel force sensors [Tra] shown in Fig. A.9. Unlike Stellar Tech sensors, these are made for the measurement of large forces up to 25000 N. Their accuracy is rated at 0.3% of the full scale output, which is thus 70 N.

A.1.3 Simulation of CDPRs in V-REP

The V-REP simulation environment [RSF13] was used in order to create a dynamic model of the previously described CDPRs including the vision sensor. Both V-REP models are shown in Fig. A.10. This gives us the capacity to use the same software to control real and virtual hardware. As a result it is possible to speed up the development, testing and debugging of algorithms in simulation (with a perfectly known ground truth), leaving only final tuning and verification for the real robots.

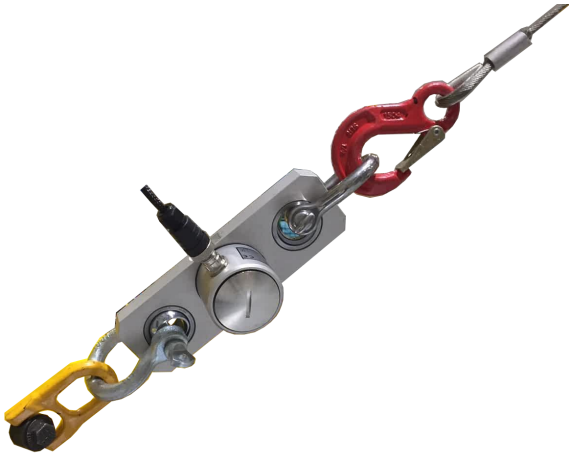
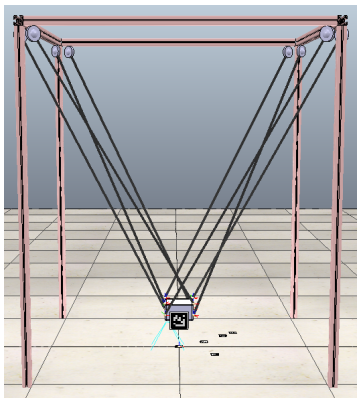
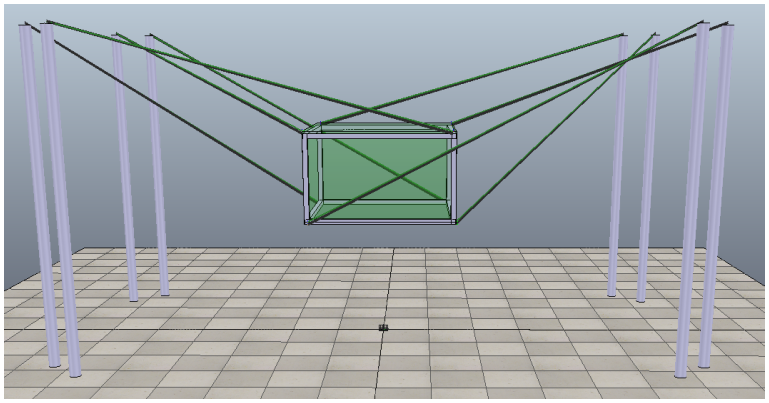


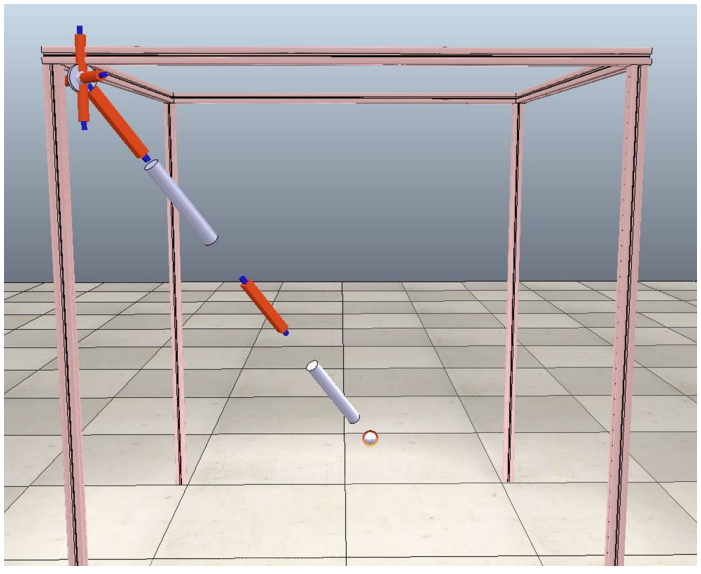
Figure A.9: Tractel force sensors



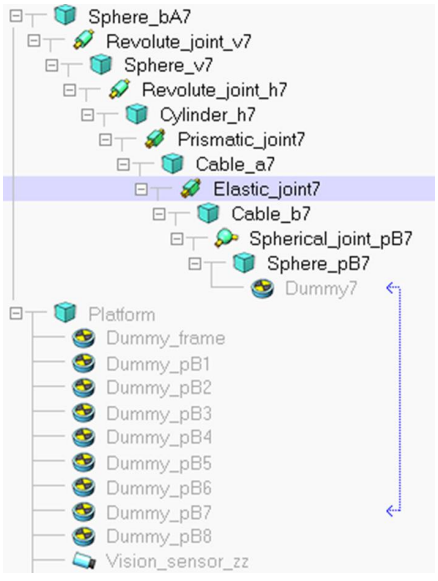
(a)



(b)



(c)



(d)

Figure A.10: V-REP model of **(a)** ACROBOT; **(b)** CAROCA. In second row: **(c)** the representation of a cable and its pulley in V-REP; **(d)** the connection of seventh cable to the moving-platform

To create a dynamical simulation, the pulleys and cables are modeled as a sequence of joints and mass objects, as shown in Fig. A.10c. The model shown in Fig. A.10 does not take into account the pulley diameter or the cable sag. Each pulley is represented as a vertical revolute passive joint followed by a small spherical mass and a horizontal revolute passive joint. The cables are modeled as a sequence of prismatic joint, cylindrical mass, prismatic joint, cylindrical mass and a final spherical joint attaches the cable to the moving-platform (Fig. A.10c). The first prismatic joint is used to change the cable length. The second prismatic joint is responsible for the cable behavior through a specific joint control callback script, which models the cable forces as either an elastic spring, when in tension, or an element transmitting zero force, when in compression.

To have a stable simulation some model design rules need to be considered [VR19]. The cable mass needs to be exaggerated given that very low mass shapes won't be able to exert large forces and that the associated joint might display soft and wobbly behavior. The parallel kinematic chains are handled in V-REP as constraints that enforce the cable free end coincidence with the associated anchor point on the moving-platform as shown in Fig. A.10d. The Vortex physical engine is used.

Simulation of cable slackness

In case of CDPR model errors or perturbations in the system some cables can become slack. It is important to be able to simulate this characteristic. For this reason, the ACROBOT model shown in Fig. A.10a is enriched with a cable slackness detection algorithm.

As described above, the second prismatic joint of the cable model, shown in Fig. A.10c, takes care of cable elasticity and transmits zero force, when it is in compression. Physically, if this joint is in compression, it means that the actual cable length is larger than the straight-line distance between cable exit point (the pulley) and cable anchor point. Thus the cable is slack.

Furthermore, it is possible to recover the amount of slack in a cable. The actual length of the cable can be retrieved via *sim.getJointPosition(name_of_the_joint)*. And then the length of the straight line between cable exit point A_i and cable anchor point B_i is computed through the Inverse Geometric Model, which is presented in Section 1.3.1.

The new model is shown in Fig. A.11. Here, the cables are in different colors to simplify the graph reading during an experiment. That is, for example, the cable $\mathcal{C}_1$ is colored blue and the corresponding curves in tension and length graphs, shown in Figs. A.11c and A.11d, respectively, are also in the same color. Furthermore, if a cable becomes slack, that is, if the elastic joint is in compression, then it is colored in red on the model (see cable $\mathcal{C}_2$ in Fig. A.11b).

It should be noted that the moving-platform pose shown in Fig. A.11a corresponds to $t = 88\,s$ on cable tension and length graphs shown in Figs. A.11c and A.11d, respectively. Similarly, the moving-platform pose shown in Fig. A.11b corresponds to $t = 94\,s$. After

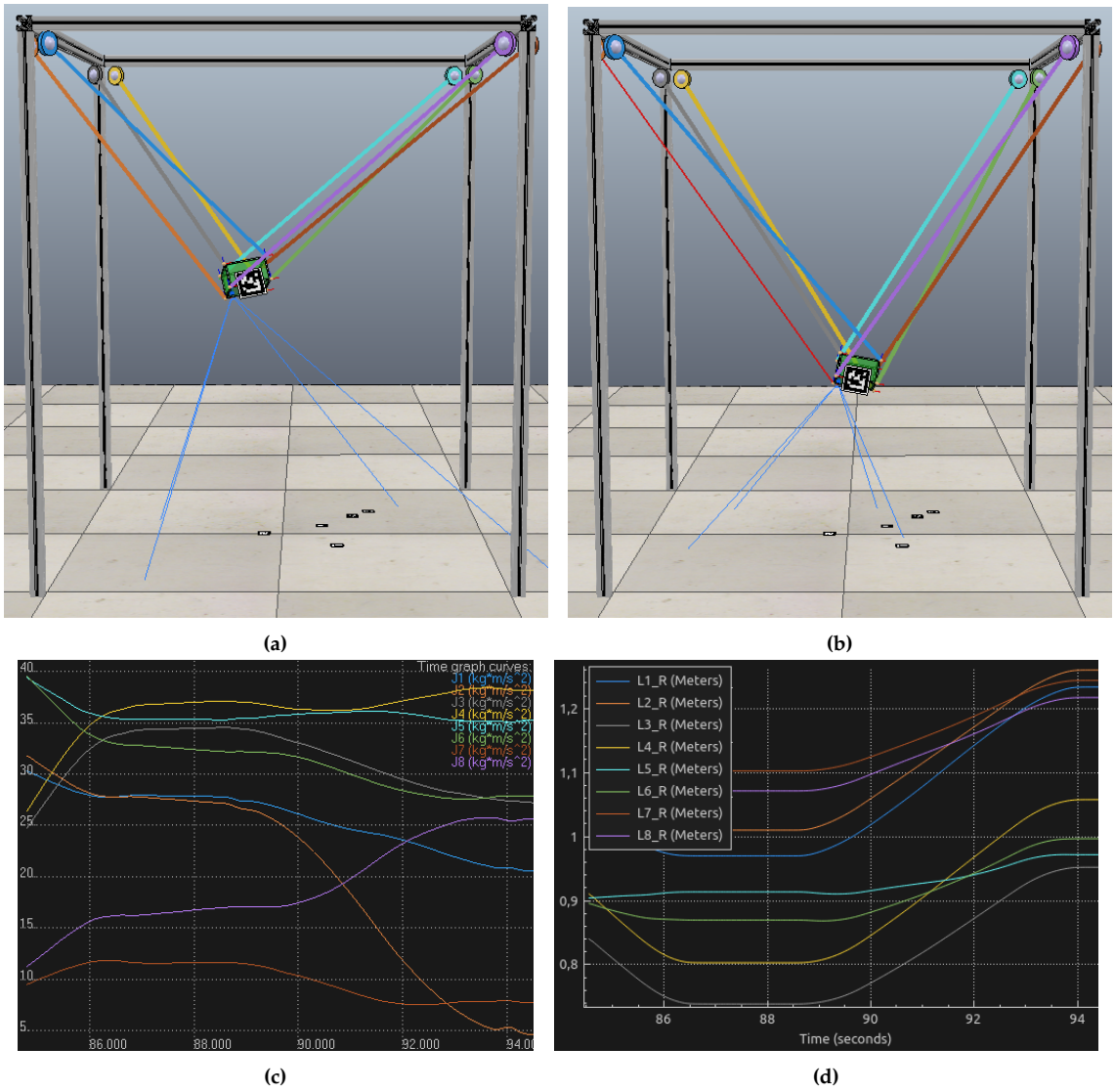


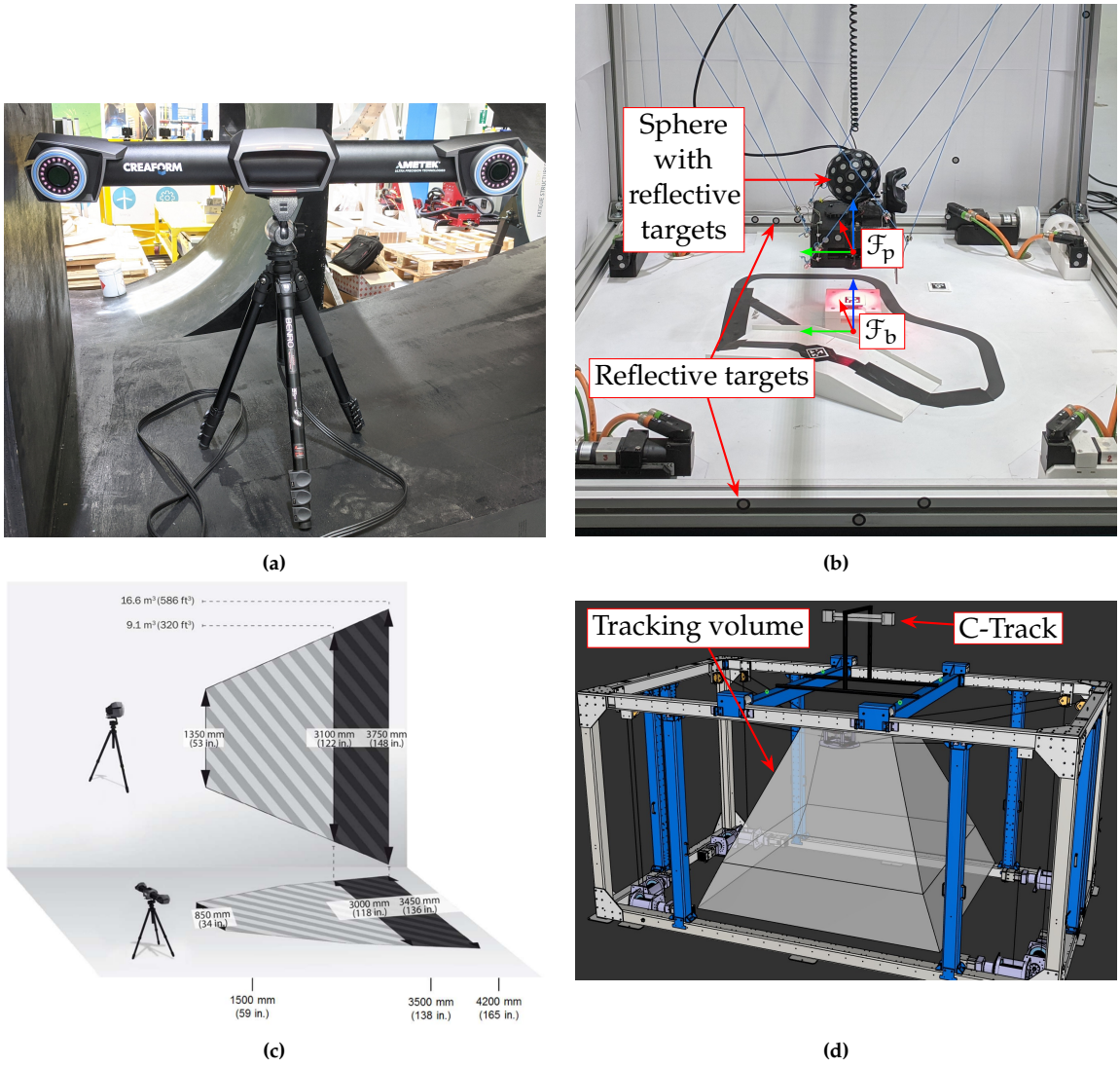
Figure A.11: New V-REP model of ACROBOT: **(a)** moving-platform in initial pose at $t = 88\,s$; **(b)** moving-platform in final pose at $t = 94\,s$; **(c)** cable tensions; **(d)** cable lengths

executing a straight-line trajectory between the first and second pose, cable $\mathcal{C}_2$ has become slack. For this reason, it is no longer colored in orange, but now has become red in Fig. A.11b.

It can be seen that at $t = 94\,s$ tension measurement of $\mathcal{C}_2$ is $\tau_2 \approx 5\,N$ instead of zero. This is due to the exaggerated cable masses that were described previously. Furthermore, the moving-platform used in this model corresponds to the small moving-platform geometry described in Section A.1.1, but its weight is also increased to 4.3 kg instead of 1.5 kg. This explains the overall increased cable tension measurements shown in Fig. A.11c.

A.2 Creaform C-Track

The Creaform measuring system is used in this thesis as ground truth for most experiments. More precisely, it is used in Chapter 2; in all experiments with CAROCA in Chapter 3; in IBVS experiments in Chapter 3; in Chapter 4 for CAROCA; in Chapter 5 for mobile object tracking experiments with ACROBOT and slackness experiments with CAROCA. Their measuring device C-Track is shown in Fig. A.12a. It is a two camera system that recognizes and localizes reflective targets, such as the ones shown in Fig. A.12b. It is possible to define a system of reflective targets as a model and to define its Cartesian frame. For example the sphere on ACROBOT moving-platform along with several reflective targets on the body of the moving-platform is defined as a single model. The origin of the moving-platform model is defined on the bottom of the moving-platform in the middle of the xy plane, as shown in Fig. A.12b. Similarly, multiple targets are placed



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

Figure A.12: Creaform measuring system: **(a)** C-Track; **(b)** ACROBOT with reflective targets; **(c)** C-Track volume

on the frame of the CDPR and the origin of base frame $\mathcal{F}_b$ is located in the middle of the ground. Having the two models, the accompanying program VXelements can track the displacement of one model with respect to the other. This measurement is only available as a CSV file after the tracking, it was thus not possible to use them directly in the control.

As can be seen in Fig. A.12c, the tracking volume is quite limited, especially the distance from C-Track. Indeed, targets can only be measured within 1.5 m and 4.2 m, thus the translation along the z axis of C-Track can be at most 2.7 m. While this is not a problem for ACROBOT, it is clear that C-Track cannot track the moving-platform of CAROCA throughout the whole workspace. To maximize the tracking volume C-Track was mounted on top of CAROCA as shown in Fig. A.12d.

Technical characteristics [CT]:

- maximum tracking volume: 16.6 m^3
- accuracy: 0.1 mm
- repeatability: 0.02 mm
- tracking frequency: up to 80 Hz
- tracking output: csv file with the Cartesian pose of the model
- minimum reflective targets per model: 4, ideally at least 6
- maximum angle of detection for targets: 45°, ideally less than that for a more precise detection

A.3 Control scheme expressed in base frame

This appendix shows in detail the expression of the control expressed in the base frame $\mathcal{F}_b$.

A.3.1 Expression of the Jacobian matrix

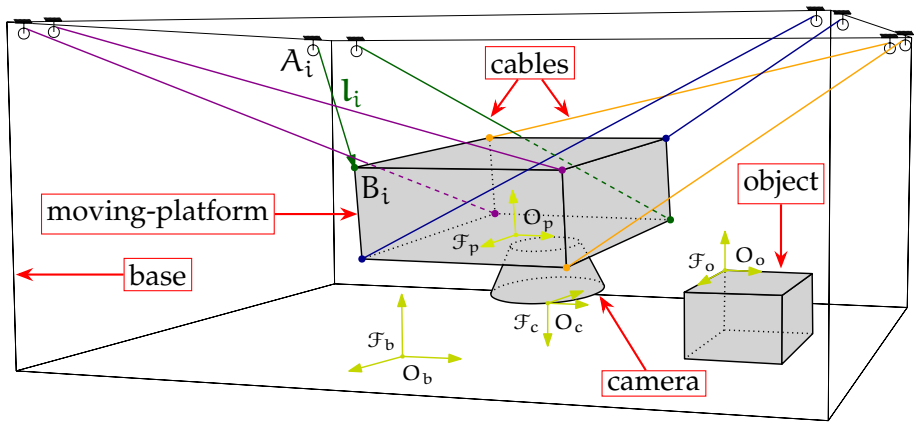


Figure A.13: Schematic of a spatial CDPR with eight cables, a camera mounted on the moving-platform and an object in the workspace

In Fig. A.13 A_i is the cable exit point, B_i is cable anchor point, l_i is the length of the ith cable, which can be defined as:

$$l_i \, {}^b\mathbf{u}_i = {}^b\overrightarrow{A_iB_i} = {}^b\mathbf{b}_i - {}^b\mathbf{a}_i = {}^b\mathbf{R}_p \, {}^p\mathbf{b}_i - {}^b\mathbf{a}_i + {}^b\mathbf{t}_p \tag{A.1}$$

where ${}^b\mathbf{u}_i$ is the unit vector of ${}^b\overrightarrow{A_iB_i}$ that is expressed as:

$${}^b\mathbf{u}_i = \frac{{}^b\overrightarrow{A_iB_i}}{\left\| {}^b\overrightarrow{A_iB_i} \right\|_2} = \frac{{}^b\mathbf{b}_i - {}^b\mathbf{a}_i}{\left\| {}^b\overrightarrow{A_iB_i} \right\|_2} = \frac{{}^b\mathbf{R}_p \, {}^p\mathbf{b}_i - {}^b\mathbf{a}_i + {}^b\mathbf{t}_p}{\left\| {}^b\overrightarrow{A_iB_i} \right\|_2} \tag{A.2}$$

Here, ${}^b\mathbf{R}_p$ and ${}^b\mathbf{t}_p$ are the rotation matrix and the translation vector of the homogeneous transformation matrix ${}^b\mathbf{T}_p$ from $\mathcal{F}_b$ to $\mathcal{F}_p$.

To obtain the cable velocities $\dot{\mathbf{l}}$, (A.1) is differentiated with respect to time:

$$\frac{d\left(l_i \, {}^b\mathbf{u}_i\right)}{dt} = \frac{d\left({}^b\overrightarrow{A_iB_i}\right)}{dt}$$

$$\dot{l}_i \, {}^b\mathbf{u}_i + l_i \, {}^b\dot{\mathbf{u}}_i = \frac{d\left({}^b\mathbf{R}_p \, {}^p\mathbf{b}_i\right)}{dt} - \frac{d\left({}^b\mathbf{a}_i\right)}{dt} + \frac{d\left({}^b\mathbf{t}_p\right)}{dt}$$

Both sides of equation are multiplied by ${}^b\mathbf{u}_i^\top$ on the left. Also ${}^b\mathbf{a}_i$ is constant, therefore its derivative is zero:

$$\dot{l}_i \, {}^b\mathbf{u}_i^\top \, {}^b\mathbf{u}_i + l_i \, {}^b\mathbf{u}_i^\top \, {}^b\dot{\mathbf{u}}_i = {}^b\mathbf{u}_i^\top \left(\frac{d\left({}^b\mathbf{t}_p\right)}{dt} + \frac{d\left({}^b\mathbf{R}_p \, {}^p\mathbf{b}_i\right)}{dt} \right)$$

Since, according to [Dal+11], ${}^b\mathbf{u}_i^\top \, {}^b\mathbf{u}_i = 1$ and ${}^b\mathbf{u}_i^\top \, {}^b\dot{\mathbf{u}}_i = 0$, then:

$$\dot{l}_i = {}^b\mathbf{u}_i^\top \left(\frac{d\left({}^b\mathbf{t}_p\right)}{dt} + \frac{d\left({}^b\mathbf{R}_p \, {}^p\mathbf{b}_i\right)}{dt} \right)$$

If we take into account that ${}^b\mathbf{R}_p \, {}^p\mathbf{b}_i = {}^b\mathbf{b}_i$, then its derivation becomes:

$$\frac{d\left({}^b\mathbf{R}_p \, {}^p\mathbf{b}_i\right)}{dt} = \frac{d\left({}^b\mathbf{b}_i\right)}{dt} = {}^b\dot{\mathbf{b}}_i = {}^b\dot{\mathbf{R}}_p \, {}^p\mathbf{b}_i = {}^b\dot{\mathbf{R}}_p \, {}^b\mathbf{R}_p^\top \, {}^b\mathbf{b}_i = [{}^b\boldsymbol{\omega}_p]_\times \, {}^b\mathbf{b}_i = {}^b\boldsymbol{\omega}_p \times {}^b\mathbf{b}_i$$

where ${}^b\boldsymbol{\omega}_p$ is the angular velocity.

Furthermore, $\frac{d\left({}^b\mathbf{t}_p\right)}{dt} = {}^b\dot{\mathbf{t}}_p = {}^b\boldsymbol{\nu}_p$ is the translational velocity of the moving-platform. Finally, the cable velocities are:

$$\dot{l}_i = {}^b\mathbf{u}_i^\top \, {}^b\boldsymbol{\nu}_p + {}^b\mathbf{u}_i^\top \left({}^b\boldsymbol{\omega}_p \times {}^b\mathbf{b}_i \right) = {}^b\mathbf{u}_i^\top \, {}^b\boldsymbol{\nu}_p + {}^b\boldsymbol{\omega}_p^\top \left({}^b\mathbf{b}_i \times {}^b\mathbf{u}_i \right)$$

And if expressed in matrix form, the cable velocity vector $\dot{\mathbf{l}}$ becomes:

$$\dot{\mathbf{l}} = {}^b\mathbf{A} \, {}^b\mathbf{v}_p \tag{A.3}$$

where the Jacobian matrix ${}^{b}\mathbf{A}$ is

$${}^{b}\mathbf{A} = \begin{bmatrix} {}^{b}\mathbf{u}_1^{\top} & \left({}^{b}\mathbf{R}_p\, {}^{p}\mathbf{b}_1 \times {}^{b}\mathbf{u}_1\right)^{\top} \\ \vdots & \vdots \\ {}^{b}\mathbf{u}_m^{\top} & \left({}^{b}\mathbf{R}_p\, {}^{p}\mathbf{b}_m \times {}^{b}\mathbf{u}_m\right)^{\top} \end{bmatrix}$$

and the Cartesian velocity ${}^{b}\mathbf{v}_p$ is

$${}^{b}\mathbf{v}_p = \begin{bmatrix} {}^{b}\boldsymbol{\nu}_p \\ {}^{b}\boldsymbol{\omega}_p \end{bmatrix}$$

A.3.2 Expression of the Adjoint matrix

This appendix shows in detail the expression of the Adjoint matrix, which is used to relate the moving-platform twist ${}^{b}\mathbf{v}_p$ and the camera velocity ${}^{c}\mathbf{v}_c$.

First, let us express ${}^{c}\mathbf{v}_c$ from ${}^{b}\mathbf{v}_p$:

$${}^{c}\boldsymbol{\nu}_c = {}^{c}\mathbf{R}_b\, {}^{b}\boldsymbol{\nu}_p - {}^{c}\mathbf{R}_b [{}^{b}\overrightarrow{PC}]_{\times}\, {}^{b}\boldsymbol{\omega}_p \tag{A.4}$$

$${}^{c}\boldsymbol{\omega}_c = {}^{c}\mathbf{R}_b\, {}^{b}\boldsymbol{\omega}_c = {}^{c}\mathbf{R}_b\, {}^{b}\boldsymbol{\omega}_p \tag{A.5}$$

The above equation (A.5) is possible, because both origins O_p and O_c of frames $\mathcal{F}_p$ and $\mathcal{F}_c$, respectively, are located on the same moving-platform, therefore it can be deduced that ${}^{b}\boldsymbol{\omega}_p = {}^{b}\boldsymbol{\omega}_c$.

Taking into account (A.4) and (A.5), it is possible to express ${}^{c}\mathbf{v}_c$ in the matrix form:

$${}^{c}\mathbf{v}_c = \begin{bmatrix} {}^{c}\boldsymbol{\nu}_c \\ {}^{c}\boldsymbol{\omega}_c \end{bmatrix} = \begin{bmatrix} {}^{c}\mathbf{R}_b & -{}^{c}\mathbf{R}_b [{}^{b}\overrightarrow{PC}]_{\times} \\ \mathbf{0}_3 & {}^{c}\mathbf{R}_b \end{bmatrix} \begin{bmatrix} {}^{b}\boldsymbol{\nu}_p \\ {}^{b}\boldsymbol{\omega}_p \end{bmatrix} = {}^{b}\mathbf{A}_d^{-1}\, {}^{b}\mathbf{v}_p \tag{A.6}$$

Here, the (6×6) matrix is denoted as ${}^{b}\mathbf{A}_d^{-1}$, it is the inverse of the Adjoint transformation matrix.

We are actually interested in the inverse relation:

$${}^{b}\mathbf{v}_p = \left({}^{b}\mathbf{A}_d^{-1}\right)^{-1} {}^{c}\mathbf{v}_c = \mathbf{A}_d\, {}^{c}\mathbf{v}_c \tag{A.7}$$

Therefore, the Adjoint transformation matrix ${}^{b}\mathbf{A}_d$ must be determined now. First, we express ${}^{b}\boldsymbol{\omega}_p$ from (A.5):

$${}^{b}\boldsymbol{\omega}_p = {}^{c}\mathbf{R}_b^{-1}\, {}^{c}\boldsymbol{\omega}_c = {}^{b}\mathbf{R}_c\, {}^{c}\boldsymbol{\omega}_c \tag{A.8}$$

Then (A.8) is injected in (A.4) and ${}^{b}\boldsymbol{v}_{p}$ can be expressed as:

$$\begin{aligned} {}^{b}\boldsymbol{v}_{p} &= {}^{c}\mathbf{R}_{b}^{-1}\left({}^{c}\boldsymbol{v}_{c} + {}^{c}\mathbf{R}_{b}[{}^{b}\overrightarrow{PC}]_{\times}\,{}^{b}\mathbf{R}_{c}\,{}^{c}\boldsymbol{\omega}_{c}\right) = \\ &= {}^{b}\mathbf{R}_{c}\,{}^{c}\boldsymbol{v}_{c} + {}^{b}\mathbf{R}_{c}\,{}^{c}\mathbf{R}_{b}[{}^{b}\overrightarrow{PC}]_{\times}\,{}^{b}\mathbf{R}_{c}\,{}^{c}\boldsymbol{\omega}_{c} = \\ &= {}^{b}\mathbf{R}_{c}\,{}^{c}\boldsymbol{v}_{c} + [{}^{b}\overrightarrow{PC}]_{\times}\,{}^{b}\mathbf{R}_{c}\,{}^{c}\boldsymbol{\omega}_{c} \end{aligned} \tag{A.9}$$

Therefore the (A.7) can be rewritten as:

$${}^{b}\mathbf{v}_{p} = {}^{b}\mathbf{A}_{d}\,{}^{c}\mathbf{v}_{c} = \begin{bmatrix} {}^{b}\mathbf{R}_{c} & [{}^{b}\overrightarrow{PC}]_{\times}\,{}^{b}\mathbf{R}_{c} \\ \mathbf{0}_{3} & {}^{b}\mathbf{R}_{c} \end{bmatrix} \begin{bmatrix} {}^{c}\boldsymbol{v}_{c} \\ {}^{c}\boldsymbol{\omega}_{c} \end{bmatrix} \tag{A.10}$$

Finally, we rewrite $[{}^{b}\overrightarrow{PC}]_{\times} = [{}^{b}\mathbf{t}_{c}]_{\times} = {}^{b}\mathbf{R}_{p}[{}^{p}\mathbf{t}_{c}]_{\times}$, therefore the final form of the Adjoint matrix is:

$${}^{b}\mathbf{A}_{d} = \begin{bmatrix} {}^{b}\mathbf{R}_{c} & {}^{b}\mathbf{R}_{p}[{}^{p}\mathbf{t}_{c}]_{\times}\,{}^{b}\mathbf{R}_{c} \\ \mathbf{0}_{3} & {}^{b}\mathbf{R}_{c} \end{bmatrix} \tag{A.11}$$

A.3.3 Closed-loop equation

Thus, we can write the full system equation from Eqs. (A.3), (1.32) and (A.10):

$$\dot{\boldsymbol{e}} = \mathbf{L}_{s}\,{}^{b}\mathbf{A}_{d}^{-1}\,{}^{b}\mathbf{A}^{\dagger}\,\dot{\mathbf{l}} \tag{A.12}$$

and upon injecting (A.10) and (1.33) into (A.3), the output of the control scheme, i.e. the cable velocity vector $\dot{\mathbf{l}}$, takes the form:

$$\dot{\mathbf{l}} = -\lambda\,{}^{b}\widehat{\mathbf{A}}\,{}^{b}\widehat{\mathbf{A}}_{d}\,\widehat{\mathbf{L}}_{s}^{-1}\,\boldsymbol{e} \tag{A.13}$$

Making the final closed-loop equation:

$$\dot{\boldsymbol{e}} = -\lambda\,\mathbf{L}_{s}\,{}^{b}\mathbf{A}_{d}^{-1}\,{}^{b}\mathbf{A}^{\dagger}\,{}^{b}\widehat{\mathbf{A}}\,{}^{b}\widehat{\mathbf{A}}_{d}\,\widehat{\mathbf{L}}_{s}^{-1}\,\boldsymbol{e} \tag{A.14}$$

It should be noted that ${}^{b}\mathbf{A}_{d}^{-1}\,{}^{b}\mathbf{A}^{\dagger} = \mathbf{A}_{d}^{-1}\mathbf{A}^{\dagger}$, where $\mathbf{A}_{d}$ and $\mathbf{A}$ are given in (1.37) and (1.19), respectively. Indeed, no matter the chosen frame, the control output will be the same. For this reason it was chosen to use the Jacobian and the Adjoint matrix expressed in $\mathcal{F}_{p}$, because they have a slightly simpler form.

A.4 Velocity Control

To compare the vision-based control of CDPRs to another existing method, it was chosen to implement a simple velocity controller. Given an initial moving-platform pose ${}^{b}\mathbf{p}_{p_0}$ and a desired pose ${}^{b}\mathbf{p}_{p^*}$, a trajectory is generated using a fifth-order

polynomial [BCG17]:

$$s = bt^5 + ct^4 + dt^3 + et^2 + ft + g \tag{A.15}$$

while considering the following conditions at the start and at the end of the trajectory:

$$\begin{cases} s(t_0) = 0 & \dot{s}(t_0) = 0 & \ddot{s}(t_0) = 0 \\ s(t_f) = 1 & \dot{s}(t_f) = 0 & \ddot{s}(t_f) = 0 \end{cases} \tag{A.16}$$

where t_0 is the initial time, which generally is $t_0 = 0\,s$; and t_f is the final time.

The moving-platform position ${}^{b}\mathbf{t}_p$ and translational velocity ${}^{b}\boldsymbol{\nu}_p$ as a function of time are expressed as:

$$\begin{cases} {}^{b}\mathbf{t}_p(t) = \mathbf{t}_s + (\mathbf{t}_f - \mathbf{t}_s)\, s(t) & \text{(A.17a)} \\ {}^{b}\boldsymbol{\nu}_p(t) = (\mathbf{t}_f - \mathbf{t}_s)\, \dot{s}(t) & \text{(A.17b)} \end{cases}$$

where ${}^{b}\mathbf{t}_{p_0}$ and ${}^{b}\mathbf{t}_{p^*}$ are the translational parts of ${}^{b}\mathbf{p}_{p_0}$ and ${}^{b}\mathbf{p}_{p^*}$, respectively.

As for the rotation, the rotational distance can be computed as:

$${}^{p_0}\mathbf{R}_{p^*} = {}^{b}\mathbf{R}_{p_0}^{\top}\, {}^{b}\mathbf{R}_{p^*} \tag{A.18}$$

where ${}^{b}\mathbf{R}_{p_0}$ and ${}^{b}\mathbf{R}_{p^*}$ are the rotation matrices for poses ${}^{b}\mathbf{p}_{p_0}$ and ${}^{b}\mathbf{p}_{p^*}$, respectively.

${}^{p_0}\mathbf{R}_{p^*}$ is then changed to axis-angle representation and it is noted as $\theta_p \mathbf{u}_p$. Here, the unit vector $\mathbf{u}_p$ is constant and the angle θ_p is a function of time $\theta_p(t) = \theta_p\, s(t)$. Once the angle $\theta_p(t)$ is calculated for time t, the corresponding rotation matrix ${}^{p_0}\mathbf{R}_{p_{curr}}$ can be computed. Consequently, the current rotation matrix of the moving-platform is computed as:

$${}^{b}\mathbf{R}_{p_{curr}} = {}^{b}\mathbf{R}_{p_0}\, {}^{p_0}\mathbf{R}_{p_{curr}} \tag{A.19}$$

Finally, the angular velocity ${}^{b}\boldsymbol{\omega}_p$ is computed as:

$${}^{b}\boldsymbol{\omega}_p(t) = \theta_p\, \mathbf{u}_p\, \dot{s}(t) \tag{A.20}$$

Then the cable velocities are computed as:

$$\dot{\mathbf{l}} = \widehat{\mathbf{A}}\, {}^{b}\mathbf{v}_p(t) \tag{A.21}$$

where ${}^{b}\mathbf{v}_p(t) = \begin{bmatrix} {}^{b}\boldsymbol{\nu}_p^{\top}(t) & {}^{b}\boldsymbol{\omega}_p^{\top}(t) \end{bmatrix}^{\top}$ is given from (A.17b) and (A.20). Note that in (A.21), the Jacobian matrix $\widehat{\mathbf{A}}$ uses the moving-platform pose estimation that is computed from (A.17a) and (A.19).

A.5 Control Stability Workspace results

All of the CSW numerical results are shown here. The tables show the change in CSW volume as a function of increase in perturbation range. The results are shown as follows:

- PBVS on a planar CDPR in Table A.8
- PBVS on ACROBOT with small moving-platform in Table A.9
- PBVS on ACROBOT with large moving-platform in Table A.10
- PBVS on ACROBOT in a fully constrained configuration with large moving-platform in Table A.11
- PBVS on CAROCA in Table A.12
- 2½D VS on ACROBOT with small moving-platform in Table A.13
- 2½D VS on ACROBOT with large moving-platform in Table A.14
- 2½D VS on CAROCA in Table A.15
- IBVS on ACROBOT with small moving-platform in Table A.16
- IBVS on ACROBOT with large moving-platform in Table A.17
- IBVS on CAROCA in Table A.18

Table A.8: Planar CDPR CSW area as a function of perturbation range

Perturb.	Range	CSW area	% of full
r_{bp}	0.03 m	0.878 m^2	95.4%
	0.05 m	0.774 m^2	84.1%
	0.10 m	0.336 m^2	36.5%
	0.15 m	0 m^2	0%

(a)

Perturb.	Range	CSW area	% of full
$\Delta\theta_{bp}$	3°	0.878 m^2	95.4%
	5°	0.869 m^2	94.5%
	10°	0.856 m^2	93.0%
	20°	0.788 m^2	85.6%
	30°	0.687 m^2	74.7%
	50°	0 m^2	0%

(b)

Perturb.	Range	CSW area	% of full
r_{pc}	0.01 m	0.878 m^2	95.4%
	0.03 m	0.859 m^2	93.4%
	0.05 m	0.858 m^2	93.3%
	0.10 m	0.855 m^2	92.9%
	0.20 m	0.849 m^2	92.3%

(c)

Perturb.	Range	CSW area	% of full
$\Delta\theta_{pc}$	3°	0.878 m^2	95.4%
	5°	0.875 m^2	95.1%
	10°	0.869 m^2	94.5%
	20°	0.861 m^2	93.6%
	30°	0.844 m^2	91.7%
	50°	0.694 m^2	75.4%

(d)

Perturb.	Range	CSW area	% of full
r_{Ai}	0.005 m	0.878 m^2	95.4%
	0.01 m	0.849 m^2	92.3%
	0.02 m	0.783 m^2	85.1%
	0.03 m	0.684 m^2	74.3%
	0.04 m	0.551 m^2	59.9%
	0.05 m	0.405 m^2	44.0%
	0.06 m	0.158 m^2	17.2%
	0.08 m	0 m^2	0%

(e)

Perturb.	Range	CSW area	% of full
r_{Bi}	0.005 m	0.878 m^2	95.4%
	0.01 m	0.849 m^2	92.3%
	0.02 m	0.790 m^2	85.9%
	0.03 m	0.729 m^2	79.2%
	0.04 m	0.578 m^2	62.8%
	0.05 m	0.387 m^2	42.1%
	0.06 m	0.121 m^2	13.2%
	0.08 m	0 m^2	0%

(f)

Perturb.	Range	CSW area	% of full
r_s	0.01 m	0.878 m^2	95.4%
	0.05 m	0.869 m^2	94.5%
	0.10 m	0.853 m^2	92.7%
	0.20 m	0.850 m^2	92.4%
	0.30 m	0.819 m^2	89.0%
	0.40 m	0.804 m^2	87.4%
	0.50 m	0.757 m^2	82.3%

(g)

Table A.9: PBVS of ACROBOT with small moving-platform

Pert.	Range	CSW volume	% of full
	0.03 m	0.875 m^3	86.3%
	0.05 m	0.828 m^3	81.6%
	0.10 m	0.733 m^3	72.3%
	0.20 m	0.597 m^3	58.9%
r_{bp}	0.25 m	0.532 m^3	52.5%
	0.30 m	0.480 m^3	47.4%
	0.50 m	0.298 m^3	29.4%
	0.60 m	0.206 m^3	20.3%
	0.70 m	0.088 m^3	8.7%

(a)

Pert.	Range	CSW volume	% of full
	3°	0.875 m^3	86.3%
	5°	0.853 m^3	84.1%
	10°	0.765 m^3	75.5%
$\Delta\theta_{bp}$	20°	0.565 m^3	55.7%
	30°	0.356 m^3	35.1%
	40°	0.077 m^3	7.6%
	50°	0 m^3	0%

(b)

Pert.	Range	CSW volume	% of full
	0.01 m	0.875 m^3	86.3%
	0.03 m	0.874 m^3	86.2%
r_{pc}	0.05 m	0.874 m^3	86.2%
	0.10 m	0.874 m^3	86.2%
	0.20 m	0.873 m^3	86.1%

(c)

Pert.	Range	CSW volume	% of full
	3°	0.875 m^3	86.3%
	5°	0.874 m^3	86.2%
	10°	0.871 m^3	85.9%
$\Delta\theta_{pc}$	20°	0.826 m^3	81.5%
	30°	0.743 m^3	73.3%
	40°	0.556 m^3	54.8%
	50°	0 m^3	0%

(d)

Pert.	Range	CSW volume	% of full
	0.005 m	0.875 m^3	86.3%
	0.01 m	0.680 m^3	67.0%
	0.02 m	0.518 m^3	51.1%
r_{Ai}	0.03 m	0.325 m^3	32.1%
	0.04 m	0.138 m^3	13.6%
	0.05 m	0.040 m^3	3.9%
	0.1 m	0 m^3	0%

(e)

Pert.	Range	CSW volume	% of full
	0.005 m	0.875 m^3	86.3%
	0.01 m	0.654 m^3	64.5%
r_{Bi}	0.02 m	0.391 m^3	38.5%
	0.03 m	0.091 m^3	8.9%
	0.04 m	0 m^3	0%

(f)

Table A.10: PBVS of ACROBOT with large moving-platform

Pert.	Range	CSW volume	% of full
	0.03 m	0.880 m^3	87.0%
	0.05 m	0.837 m^3	82.5%
	0.10 m	0.742 m^3	73.2%
	0.20 m	0.599 m^3	59.1%
r_{bp}	0.25 m	0.540 m^3	53.2%
	0.30 m	0.490 m^3	48.3%
	0.50 m	0.295 m^3	29.1%
	0.60 m	0.201 m^3	19.8%
	0.70 m	0.109 m^3	10.8%

(a)

Pert.	Range	CSW volume	% of full
	3°	0.880 m^3	87.0%
	5°	0.856 m^3	84.5%
	10°	0.765 m^3	75.5%
$\Delta\theta_{bp}$	20°	0.579 m^3	57.1%
	30°	0.381 m^3	37.6%
	40°	0.111 m^3	10.9%
	50°	0 m^3	0%

(b)

Pert.	Range	CSW volume	% of full
	0.01 m	0.880 m^3	87.0%
	0.03 m	0.879 m^3	86.7%
r_{pc}	0.05 m	0.879 m^3	86.7%
	0.10 m	0.879 m^3	86.7%
	0.20 m	0.878 m^3	86.6%

(c)

Pert.	Range	CSW volume	% of full
	3°	0.880 m^3	87.0%
	5°	0.878 m^3	86.6%
	10°	0.875 m^3	86.3%
$\Delta\theta_{pc}$	20°	0.861 m^3	84.9%
	30°	0.767 m^3	75.6%
	40°	0.606 m^3	59.8%
	50°	0 m^3	0%

(d)

Pert.	Range	CSW volume	% of full
	0.005 m	0.880 m^3	87.0%
	0.01 m	0.762 m^3	75.2%
	0.02 m	0.676 m^3	66.7%
r_{Ai}	0.03 m	0.572 m^3	56.4%
	0.04 m	0.461 m^3	45.5%
	0.05 m	0.348 m^3	34.3%
	0.1 m	0.003 m^3	0.3%

(e)

Pert.	Range	CSW volume	% of full
	0.005 m	0.880 m^3	87.0%
	0.01 m	0.754 m^3	74.3%
	0.02 m	0.618 m^3	61.0%
r_{Bi}	0.03 m	0.447 m^3	44.0%
	0.04 m	0.244 m^3	24.1%
	0.05 m	0.070 m^3	6.9%
	0.1 m	0 m^3	0%

(f)

Table A.11: PBVS of ACROBOT in fully constrained configuration

Pert.	Range	CSW volume	% of full
	0.03 m	0.985 m^3	97.0%
	0.05 m	0.917 m^3	90.4%
	0.10 m	0.348 m^3	34.3%
	0.20 m	0.139 m^3	13.8%
r_{bp}	0.25 m	0.050 m^3	4.9%
	0.30 m	0.0 m^3	0%

(a)

Pert.	Range	CSW volume	% of full
	3°	0.985 m^3	97.0%
	5°	0.983 m^3	96.9%
$\Delta\theta_{bp}$	10°	0.945 m^3	93.2%
	20°	0.461 m^3	45.4%
	30°	0.366 m^3	36.1%
	40°	0.166 m^3	16.40%
	50°	0.017 m^3	1.7%

(b)

Pert.	Range	CSW volume	% of full
	0.01 m	0.985 m^3	97.0%
	0.03 m	0.984 m^3	97.0%
r_{pc}	0.05 m	0.984 m^3	97.0%
	0.10 m	0.984 m^3	97.0%
	0.20 m	0.983 m^3	97.0%

(c)

Pert.	Range	CSW volume	% of full
	3°	0.985 m^3	97.0%
	5°	0.985 m^3	97.0%
$\Delta\theta_{pc}$	10°	0.984 m^3	97.0%
	20°	0.983 m^3	96.9%
	30°	0.972 m^3	95.8%
	40°	0.890 m^3	87.8%
	50°	0.720 m^3	71.0%

(d)

Pert.	Range	CSW volume	% of full
	0.005 m	0.985 m^3	97.0%
	0.01 m	0.958 m^3	94.4%
r_{Ai}	0.02 m	0.896 m^3	88.4%
	0.03 m	0.698 m^3	68.9%
	0.04 m	0.548 m^3	54.1%
	0.05 m	0.479 m^3	47.3%
	0.1 m	0.222 m^3	21.9%

(e)

Pert.	Range	CSW volume	% of full
	0.005 m	0.985 m^3	97.0%
r_{Bi}	0.01 m	0.956 m^3	94.3%
	0.02 m	0.883 m^3	87.1%
	0.03 m	0.673 m^3	66.4%
	0.04 m	0.593 m^3	58.5%
	0.05 m	0.529 m^3	52.1%
	0.1 m	0.0 m^3	0%

(f)

Table A.12: PBVS of CAROCA

Pert.	Range	CSW volume	% of full
r_{bp}	0.03 m	63.064 m^3	76.6%
	0.05 m	59.543 m^3	72.3%
	0.10 m	53.046 m^3	64.4%
	0.20 m	43.085 m^3	52.3%
	0.25 m	39.674 m^3	48.2%
	0.30 m	35.827 m^3	43.5%
	0.50 m	24.450 m^3	29.7%
	0.60 m	19.974 m^3	24.3%
	0.70 m	15.825 m^3	19.2%
	0.80 m	12.257 m^3	14.9%
	0.90 m	9.366 m^3	11.4%
	1.00 m	6.898 m^3	8.4%

(a)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{bp}$	3°	63.064 m^3	76.6%
	5°	55.615 m^3	67.6%
	10°	34.657 m^3	42.1%
	20°	7.320 m^3	8.9%
	30°	0.003 m^3	0.003%
	40°	0 m^3	0%

(b)

Pert.	Range	CSW volume	% of full
r_{pc}	0.01 m	63.064 m^3	76.6%
	0.03 m	63.015 m^3	76.5%
	0.05 m	62.952 m^3	76.5%
	0.10 m	62.271 m^3	75.6%
	0.20 m	61.982 m^3	75.3%
	0.30 m	60.867 m^3	73.9%
	0.40 m	60.384 m^3	73.3%
	0.50 m	59.215 m^3	71.9%

(c)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{pc}$	3°	63.064 m^3	76.6%
	5°	62.197 m^3	75.55%
	10°	60.028 m^3	72.9%
	20°	35.772 m^3	43.5%
	30°	5.64 m^3	6.8%
	40°	0 m^3	0%

(d)

Pert.	Range	CSW volume	% of full
r_{Ai}	0.005 m	63.064 m^3	76.6%
	0.01 m	60.539 m^3	73.5%
	0.02 m	58.781 m^3	71.4%
	0.03 m	56.858 m^3	69.1%
	0.04 m	54.654 m^3	66.4%
	0.05 m	52.32 m^3	63.6%
	0.1 m	40.207 m^3	48.8%
	0.2 m	3.795 m^3	4.6%
	0.3 m	0 m^3	0%

(e)

Pert.	Range	CSW volume	% of full
r_{Bi}	0.005 m	63.064 m^3	76.6%
	0.01 m	60.141 m^3	73.0%
	0.02 m	57.555 m^3	69.9%
	0.03 m	54.486 m^3	66.2%
	0.04 m	51.181 m^3	62.2%
	0.05 m	46.580 m^3	56.6%
	0.1 m	24.926 m^3	30.3%
	0.2 m	17.187 m^3	20.9%
	0.3 m	7.920 m^3	9.6%
	0.4 m	3.684 m^3	4.5%

(f)

Table A.13: 2½D VS of ACROBOT with small moving-platform

Pert.	Range	CSW volume	% of full
r_{bp}	0.03 m	0.908 m^3	89.6%
	0.05 m	0.883 m^3	87.1%
	0.10 m	0.789 m^3	77.8%
	0.20 m	0.606 m^3	59.8%
	0.25 m	0.514 m^3	50.7%
	0.30 m	0.430 m^3	42.4%
	0.50 m	0.101 m^3	9.9%
	0.60 m	0 m^3	0%

(a)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{bp}$	3°	0.908 m^3	89.6%
	5°	0.893 m^3	88.1%
	10°	0.828 m^3	81.6%
	20°	0.502 m^3	49.5%
	30°	0 m^3	0%

(b)

Pert.	Range	CSW volume	% of full
r_{pc}	0.01 m	0.908 m^3	89.6%
	0.03 m	0.906 m^3	89.3%
	0.05 m	0.904 m^3	89.1%
	0.10 m	0.894 m^3	88.2%
	0.20 m	0.889 m^3	87.7%

(c)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{pc}$	3°	0.908 m^3	89.6%
	5°	0.907 m^3	89.4%
	10°	0.904 m^3	89.1%
	20°	0.889 m^3	87.8%
	30°	0.846 m^3	83.4%
	40°	0.603 m^3	59.4%
	50°	0 m^3	0%

(d)

Pert.	Range	CSW volume	% of full
r_{Ai}	0.005 m	0.908 m^3	89.6%
	0.01 m	0.835 m^3	82.4%
	0.02 m	0.747 m^3	73.7%
	0.03 m	0.654 m^3	64.5%
	0.04 m	0.565 m^3	55.7%
	0.05 m	0.482 m^3	47.6%
	0.1 m	0 m^3	0%

(e)

Pert.	Range	CSW volume	% of full
r_{Bi}	0.005 m	0.908 m^3	89.6%
	0.01 m	0.777 m^3	76.6%
	0.02 m	0.147 m^3	14.5%
	0.03 m	0 m^3	0%

(f)

Table A.14: 2½D VS of ACROBOT with large moving-platform

Pert.	Range	CSW volume	% of full
r_{bp}	0.03 m	0.911 m^3	89.9%
	0.05 m	0.887 m^3	87.4%
	0.10 m	0.791 m^3	78.0%
	0.20 m	0.608 m^3	60.0%
	0.25 m	0.518 m^3	51.1%
	0.30 m	0.445 m^3	43.9%
	0.50 m	0.112 m^3	11.1%
	0.60 m	0 m^3	0%

(a)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{bp}$	3°	0.911 m^3	89.9%
	5°	0.896 m^3	88.4%
	10°	0.830 m^3	81.8%
	20°	0.536 m^3	52.9%
	30°	0.0003 m^3	0.03%
	40°	0 m^3	0%

(b)

Pert.	Range	CSW volume	% of full
r_{pc}	0.01 m	0.911 m^3	89.9%
	0.03 m	0.909 m^3	89.6%
	0.05 m	0.907 m^3	89.4%
	0.10 m	0.898 m^3	88.6%
	0.20 m	0.890 m^3	87.8%

(c)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{pc}$	3°	0.911 m^3	89.9%
	5°	0.910 m^3	89.8%
	10°	0.907 m^3	89.5%
	20°	0.893 m^3	88.1%
	30°	0.851 m^3	83.9%
	40°	0.602 m^3	59.4%
	50°	0 m^3	0%

(d)

Pert.	Range	CSW volume	% of full
r_{Ai}	0.005 m	0.911 m^3	89.9%
	0.01 m	0.881 m^3	86.9%
	0.02 m	0.840 m^3	82.9%
	0.03 m	0.788 m^3	77.7%
	0.04 m	0.742 m^3	73.2%
	0.05 m	0.680 m^3	67.1%
	0.1 m	0.407 m^3	40.2%

(e)

Pert.	Range	CSW volume	% of full
r_{Bi}	0.005 m	0.911 m^3	89.9%
	0.01 m	0.867 m^3	85.5%
	0.02 m	0.674 m^3	66.5%
	0.03 m	0.209 m^3	20.6%
	0.04 m	0 m^3	0%

(f)

Table A.15: 2½D VS of CAROCA

Pert.	Range	CSW volume	% of full
r_{bp}	0.03 m	77.414 m^3	94.0%
	0.05 m	75.640 m^3	91.9%
	0.10 m	75.479 m^3	91.7%
	0.20 m	66.018 m^3	80.2%
	0.25 m	64.807 m^3	78.7%
	0.30 m	63.462 m^3	77.1%
	0.50 m	53.715 m^3	65.2%
	0.60 m	50.044 m^3	60.8%
	0.70 m	45.168 m^3	54.9%
	0.80 m	40.074 m^3	48.7%
	0.90 m	34.543 m^3	41.9%
	1.00 m	28.976 m^3	35.2%

(a)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{bp}$	3°	77.414 m^3	94.0%
	5°	75.786 m^3	92.1%
	10°	66.749 m^3	81.1%
	20°	35.205 m^3	42.8%
	30°	1.938 m^3	2.4%
	40°	0 m^3	0%

(b)

Pert.	Range	CSW volume	% of full
r_{pc}	0.01 m	77.414 m^3	94.0%
	0.03 m	77.414 m^3	94.0%
	0.05 m	77.414 m^3	94.0%
	0.10 m	77.414 m^3	94.0%
	0.20 m	77.414 m^3	94.0%
	0.30 m	62.865 m^3	76.4%
	0.40 m	0.0 m^3	0%

(c)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{pc}$	3°	77.414 m^3	94.0%
	5°	75.870 m^3	92.2%
	10°	75.866 m^3	92.1%
	20°	75.479 m^3	91.7%
	30°	75.003 m^3	91.1%
	40°	61.736 m^3	74.9%
	50°	0 m^3	0%

(d)

Pert.	Range	CSW volume	% of full
r_{Ai}	0.005 m	77.414 m^3	94.0%
	0.02 m	77.414 m^3	94.0%
	0.05 m	77.414 m^3	94.0%
	0.10 m	69.626 m^3	84.6%
	0.20 m	6.955 m^3	8.4%
	0.30 m	0.0 m^3	0%

(e)

Pert.	Range	CSW volume	% of full
r_{Bi}	0.005 m	77.414 m^3	94.0%
	0.02 m	73.931 m^3	89.8%
	0.05 m	72.614 m^3	88.2%
	0.10 m	67.191 m^3	81.6%
	0.20 m	65.308 m^3	79.3%
	0.30 m	60.751 m^3	73.8%
	0.40 m	52.703 m^3	64.0%

(f)

Table A.16: IBVS of ACROBOT with small moving-platform

Pert.	Range	CSW volume	% of full
	0.03 m	0.917 m^3	90.4%
	0.05 m	0.898 m^3	88.6%
	0.10 m	0.810 m^3	79.9%
	0.20 m	0.700 m^3	69.0%
r_{bp}	0.25 m	0.649 m^3	64.0%
	0.30 m	0.596 m^3	58.8%
	0.50 m	0.394 m^3	38.9%
	0.60 m	0.300 m^3	29.6%
	0.70 m	0.205 m^3	20.2%

(a)

Pert.	Range	CSW volume	% of full
	3°	0.917 m^3	90.4%
	5°	0.909 m^3	89.6%
	10°	0.885 m^3	87.3%
$\Delta\theta_{bp}$	20°	0.760 m^3	75.0%
	30°	0.613 m^3	60.4%
	40°	0.335 m^3	33.0%
	50°	0.03 m^3	3.0%

(b)

Pert.	Range	CSW volume	% of full
	0.01 m	0.917 m^3	90.4%
	0.03 m	0.917 m^3	90.4%
r_{pc}	0.05 m	0.917 m^3	90.4%
	0.10 m	0.916 m^3	90.4%
	0.20 m	0.916 m^3	90.3%

(c)

Pert.	Range	CSW volume	% of full
	3°	0.917 m^3	90.4%
	5°	0.917 m^3	90.4%
	10°	0.916 m^3	90.3%
$\Delta\theta_{pc}$	20°	0.913 m^3	90.0%
	30°	0.912 m^3	89.9%
	40°	0.907 m^3	89.4%
	50°	0.897 m^3	88.5%

(d)

Pert.	Range	CSW volume	% of full
	0.005 m	0.917 m^3	90.4%
	0.01 m	0.878 m^3	86.6%
	0.02 m	0.805 m^3	79.4%
r_{Ai}	0.03 m	0.718 m^3	70.8%
	0.04 m	0.643 m^3	63.4%
	0.05 m	0.570 m^3	56.2%
	0.1 m	0.002 m^3	0.2%

(e)

Pert.	Range	CSW volume	% of full
	0.005 m	0.917 m^3	90.4%
	0.01 m	0.872 m^3	86.0%
	0.02 m	0.730 m^3	71.9%
r_{Bi}	0.03 m	0.435 m^3	42.9%
	0.04 m	0.077 m^3	7.5%
	0.05 m	0 m^3	0%

(f)

Table A.17: IBVS of ACROBOT with large moving-platform

Pert.	Range	CSW volume	% of full
	0.03 m	0.918 m^3	90.5%
	0.05 m	0.900 m^3	88.8%
	0.10 m	0.816 m^3	80.5%
	0.20 m	0.703 m^3	69.3%
r_{bp}	0.25 m	0.661 m^3	65.2%
	0.30 m	0.598 m^3	58.9%
	0.50 m	0.395 m^3	38.9%
	0.60 m	0.299 m^3	29.4%
	0.70 m	0.204 m^3	20.1%

(a)

Pert.	Range	CSW volume	% of full
	3°	0.918 m^3	90.5%
	5°	0.911 m^3	89.9%
	10°	0.885 m^3	87.3%
$\Delta\theta_{bp}$	20°	0.761 m^3	75.0%
	30°	0.625 m^3	61.7%
	40°	0.403 m^3	39.8%
	50°	0.057 m^3	5.6%

(b)

Pert.	Range	CSW volume	% of full
	0.01 m	0.918 m^3	90.5%
	0.03 m	0.918 m^3	90.5%
r_{pc}	0.05 m	0.918 m^3	90.5%
	0.10 m	0.918 m^3	90.5%
	0.20 m	0.917 m^3	90.4%

(c)

Pert.	Range	CSW volume	% of full
	3°	0.918 m^3	90.5%
	5°	0.918 m^3	90.5%
	10°	0.918 m^3	90.5%
$\Delta\theta_{pc}$	20°	0.915 m^3	90.3%
	30°	0.913 m^3	90.0%
	40°	0.908 m^3	89.6%
	50°	0.900 m^3	88.8%

(d)

Pert.	Range	CSW volume	% of full
	0.005 m	0.918 m^3	90.5%
	0.01 m	0.895 m^3	88.2%
	0.02 m	0.882 m^3	86.9%
r_{Ai}	0.03 m	0.818 m^3	80.7%
	0.04 m	0.766 m^3	75.5%
	0.05 m	0.718 m^3	70.8%
	0.1 m	0.481 m^3	47.4%

(e)

Pert.	Range	CSW volume	% of full
	0.005 m	0.918 m^3	90.5%
	0.01 m	0.897 m^3	88.4%
	0.02 m	0.856 m^3	84.4%
r_{Bi}	0.03 m	0.757 m^3	74.6%
	0.04 m	0.611 m^3	60.3%
	0.05 m	0.421 m^3	41.5%
	0.1 m	0 m^3	0%

(f)

Table A.18: IBVS of CAROCA

Pert.	Range	CSW volume	% of full
r_{bp}	0.03 m	77.414 m^3	94.029%
	0.05 m	75.479 m^3	91.7%
	0.10 m	73.974 m^3	89.9%
	0.20 m	71.828 m^3	87.2%
	0.25 m	68.358 m^3	83.0%
	0.30 m	66.722 m^3	81.0%
	0.50 m	61.312 m^3	74.5%
	0.60 m	57.467 m^3	69.8%
	0.70 m	55.404 m^3	67.3%
	0.80 m	50.159 m^3	60.9%
	0.90 m	48.465 m^3	58.9%
	1.00 m	46.335 m^3	56.3%

(a)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{bp}$	3°	77.414 m^3	94.029%
	5°	75.479 m^3	91.7%
	10°	72.972 m^3	88.6%
	20°	64.107 m^3	77.9%
	30°	45.333 m^3	55.1%
	40°	22.250 m^3	27.0%
	50°	3.754 m^3	4.6%

(b)

Pert.	Range	CSW volume	% of full
r_{pc}	0.01 m	77.414 m^3	94.029%
	0.03 m	77.414 m^3	94.029%
	0.05 m	77.414 m^3	94.029%
	0.10 m	77.414 m^3	94.029%
	0.20 m	77.414 m^3	94.029%
	0.30 m	77.414 m^3	94.029%
	0.40 m	77.414 m^3	94.029%
	0.50 m	77.414 m^3	94.029%

(c)

Pert.	Range	CSW volume	% of full
$\Delta\theta_{pc}$	3°	77.414 m^3	94.029%
	5°	77.414 m^3	94.029%
	10°	77.414 m^3	94.029%
	20°	77.414 m^3	94.029%
	30°	77.414 m^3	94.029%
	40°	75.694 m^3	91.9%
	50°	75.479 m^3	91.7%

(d)

Pert.	Range	CSW volume	% of full
r_{Ai}	0.005 m	77.414 m^3	94.029%
	0.01 m	77.414 m^3	94.029%
	0.02 m	74.738 m^3	90.8%
	0.05 m	73.974 m^3	89.9%
	0.1 m	72.567 m^3	88.1%
	0.2 m	71.380 m^3	86.7%
	0.3 m	67.657 m^3	82.2%
	0.4 m	61.057 m^3	74.2%

(e)

Pert.	Range	CSW volume	% of full
r_{Bi}	0.005 m	77.414 m^3	94.029%
	0.01 m	74.738 m^3	90.8%
	0.02 m	73.974 m^3	89.9%
	0.05 m	73.544 m^3	89.3%
	0.1 m	71.456 m^3	86.8%
	0.2 m	67.416 m^3	81.9%
	0.3 m	64.163 m^3	77.9%
	0.4 m	42.427 m^3	51.5%

(f)

A.6 Additional IBVS Experiments with Different Perturbations

The experimental setup is as defined in Section 3.4.5. However, here the perturbations are added to different parameters: perturbation on camera pose in the moving-platform frame in Section A.6.1; perturbation on cable anchor points in Section A.6.2; perturbation on cable exit points in Section A.6.3.

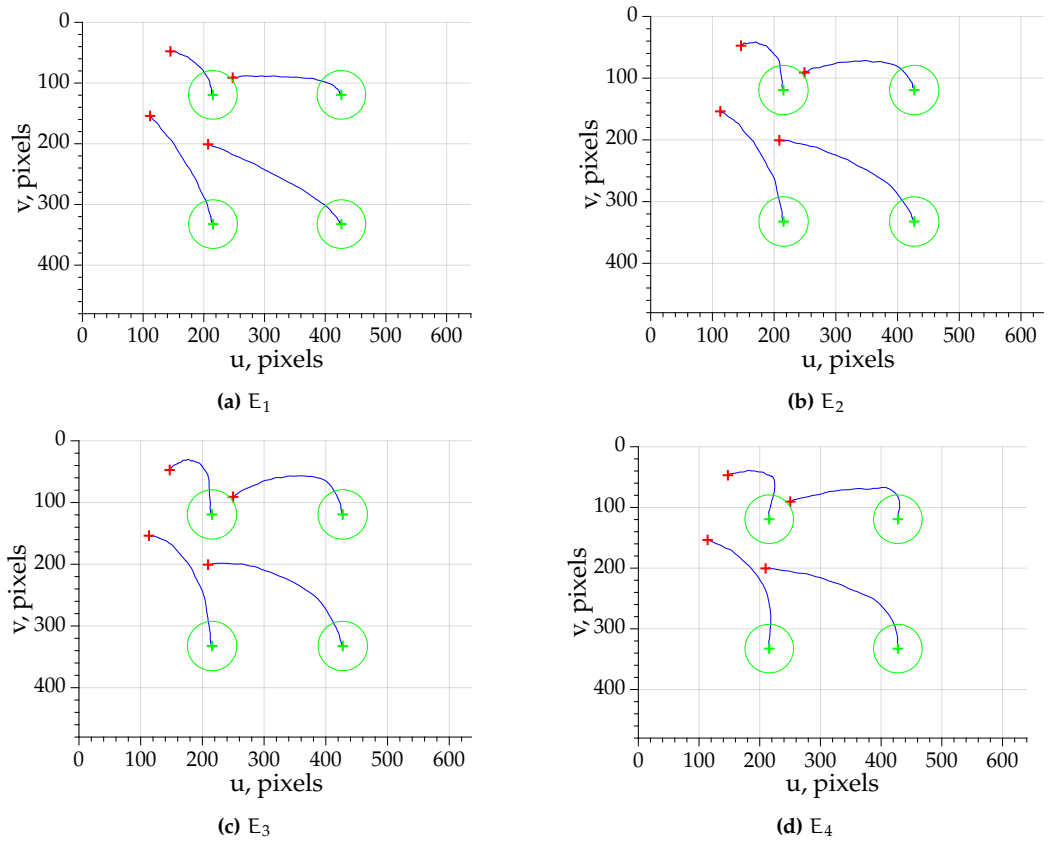
A.6.1 Perturbation on camera pose in the moving-platform frame

During stability analysis it was found that the real camera position on the moving-platform could be unknown, and some arbitrary position can be given to the controller. Here we test exactly that.

Several experiments are done with the following perturbations:

- E_1 : no perturbation added, ${}^{p}\mathbf{t}_c = \left[-0.0631\,\text{m};\ 0.0\,\text{m};\ 0.01\,\text{m}\right]$
- E_2 : $r_{pc} = 0.064\,\text{m}$ and ${}^{p}\hat{\mathbf{t}}_c = \left[0.0\,\text{m};\ 0.0\,\text{m};\ 0.0\,\text{m}\right]$;
- E_3 : $r_{pc} = 0.128\,\text{m}$ and ${}^{p}\hat{\mathbf{t}}_c = \left[0.0631\,\text{m};\ 0.0\,\text{m};\ -0.01\,\text{m}\right]$;
- E_4 : $r_{pc} = 0.106\,\text{m}$ and ${}^{p}\hat{\mathbf{t}}_c = \left[0.0\,\text{m};\ -0.08\,\text{m};\ -0.02\,\text{m}\right]$.

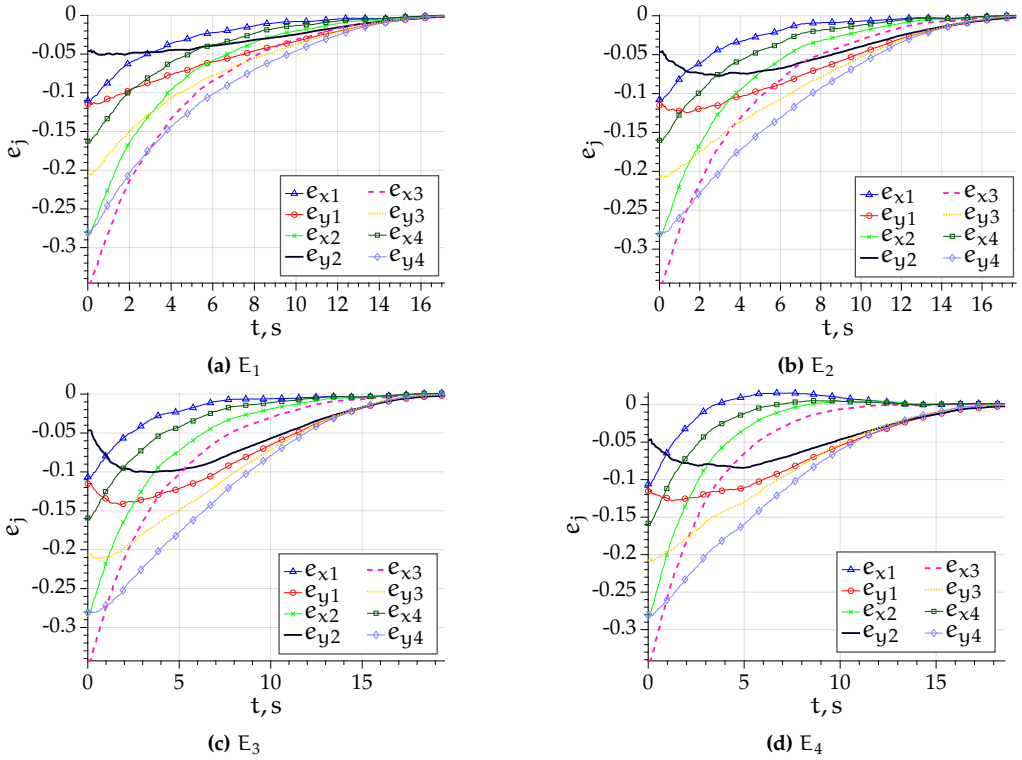
The experimental results are shown in Figs. A.14 through A.17.



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

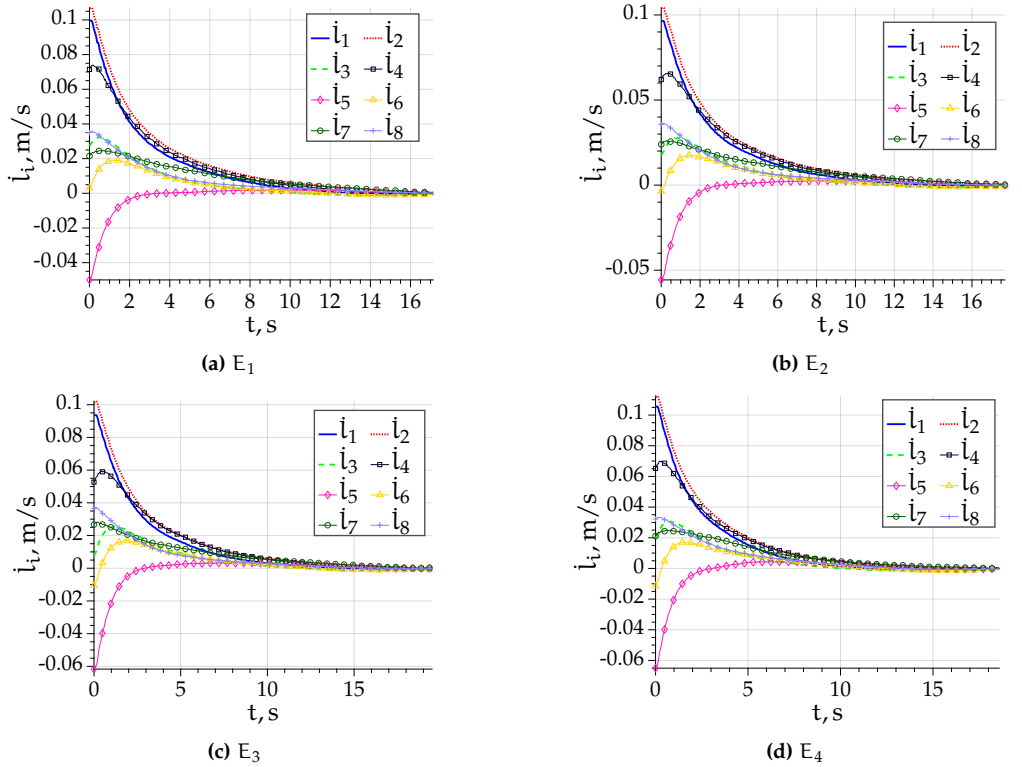
Figure A.14: Trajectory of four points in the image



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

Figure A.15: Error $\boldsymbol{e}$ over time



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

Figure A.16: Cable velocities over time

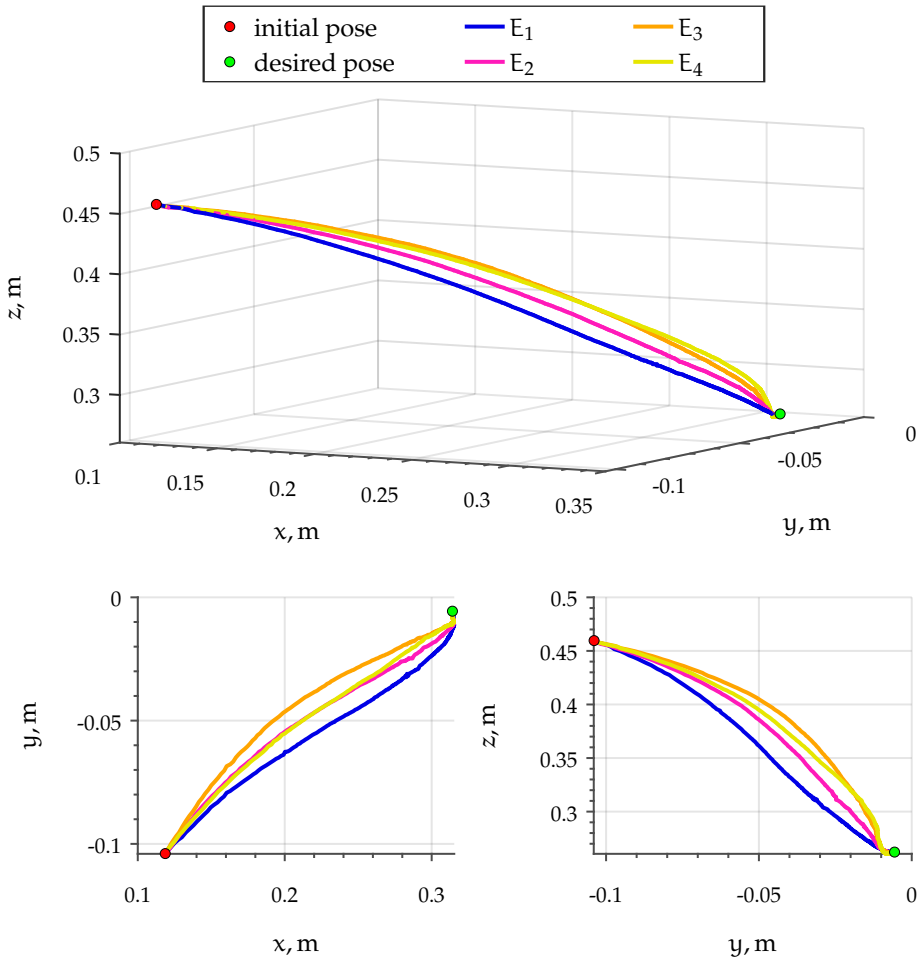


Figure A.17: Moving-platform trajectory

Of course, when no perturbation on camera position in the moving-platform frame exists, then the overall behavior is better and the error $\boldsymbol{e}$ rapidly converges to zero. Interestingly, with all three perturbation sets in E_2 to E_4, the behavior is very similar. Indeed, the direction of the deviation of the point trajectories is the same, as shown in Fig. A.14. Similarly, in all three experiments errors e_{y1} and e_{y2} increase at first and then proceed to decrease rapidly to zero. The remaining curves for components of $\boldsymbol{e}$ are also similar in these three experiments, with the exception that e_{x1} changes its sign in E_4 before converging to zero.

Thus, indeed, the camera position in the moving-platform frame can be set arbitrarily. None of the tested perturbations could make the system unstable. Moreover, the effect of these perturbations on the system is small.

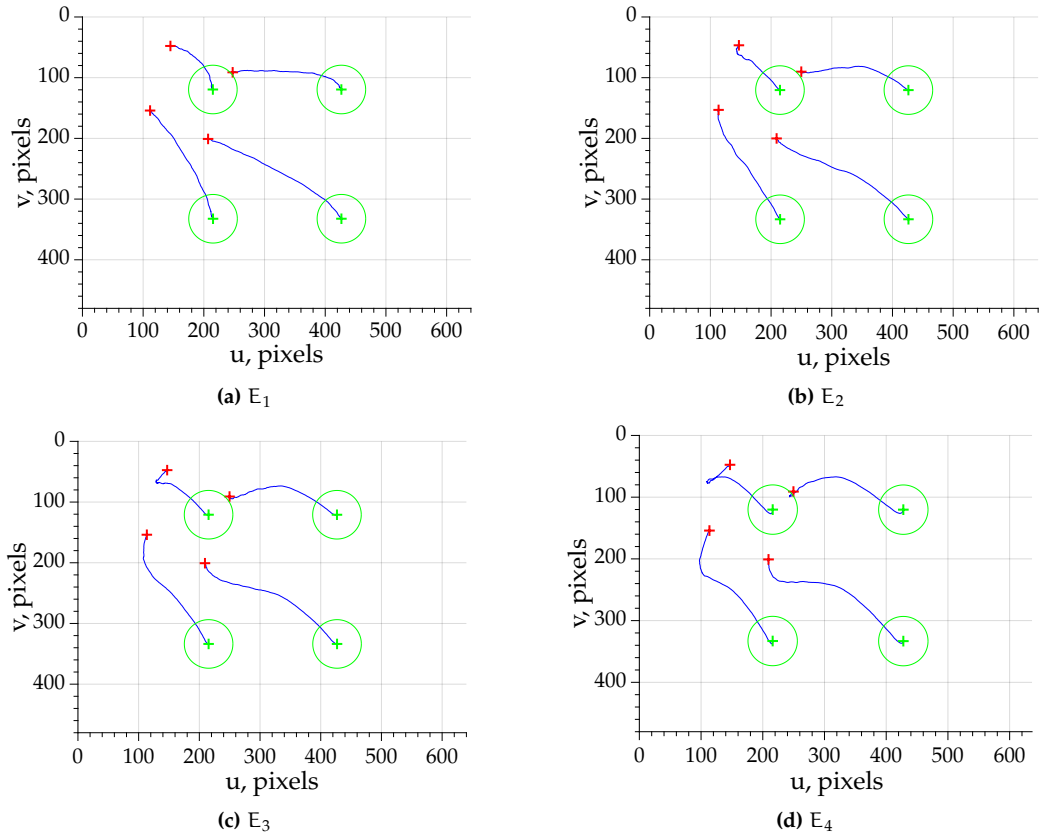
A.6.2 Perturbation on Cable Anchor Point Coordinates

In this section, perturbations are added to cable anchor point coordinates. The experiments are defined as follows:

- E_1 : no perturbation added;

- $E_2 : r_{Bi} = 0.038\,\text{m}$;
- $E_3 : r_{Bi} = 0.073\,\text{m}$;
- $E_4 : r_{Bi} = 0.108\,\text{m}$.

The experimental results are shown in Figs. A.18 through A.21.



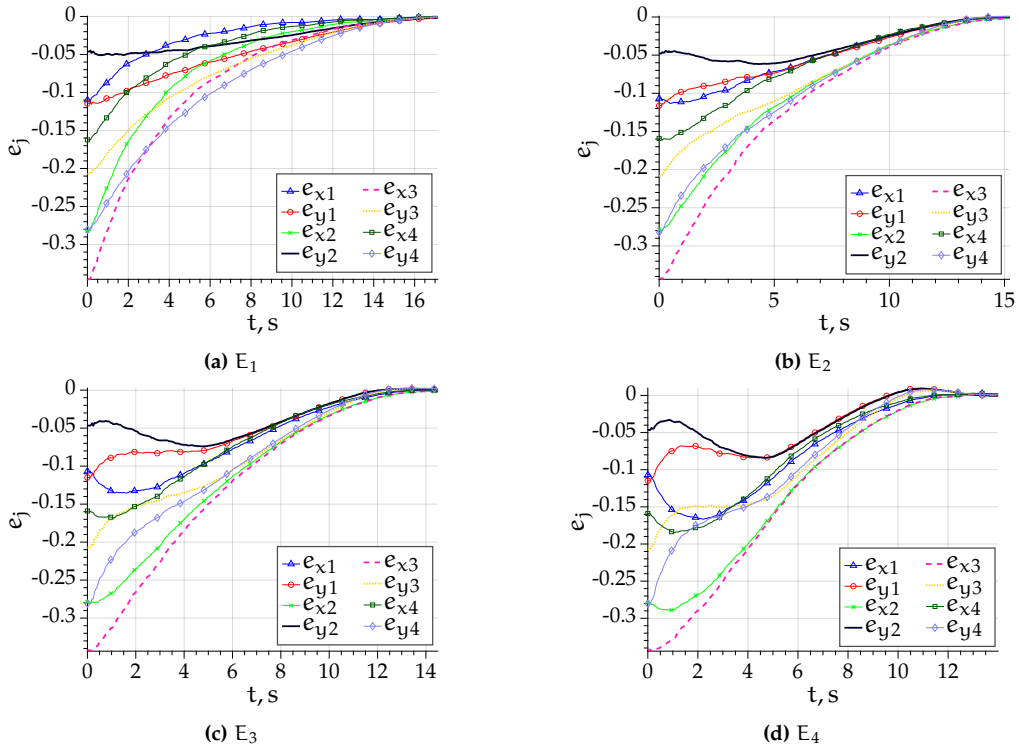
(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

Figure A.18: Trajectory of four points in the image

As can be seen in Fig. A.18, the larger the perturbation, the larger the deviation of point trajectories. The gradual change of the robot behavior can also be observed in Fig. A.21, where the moving-platform pose trajectory becomes more and more perturbed. The error $\boldsymbol{e}$ does not reduce smoothly. Indeed, its components, such as e_{x1} and e_{x4} increase before decreasing to zero. Moreover, there are components, such as e_{y2} that start to decrease, then increase even above its initial value and only then do they converge to zero. Interestingly, the trajectory time decreases. Indeed, without perturbations in E_1 the task takes about 17 s, while in E_4 it decreases to about 14 s.

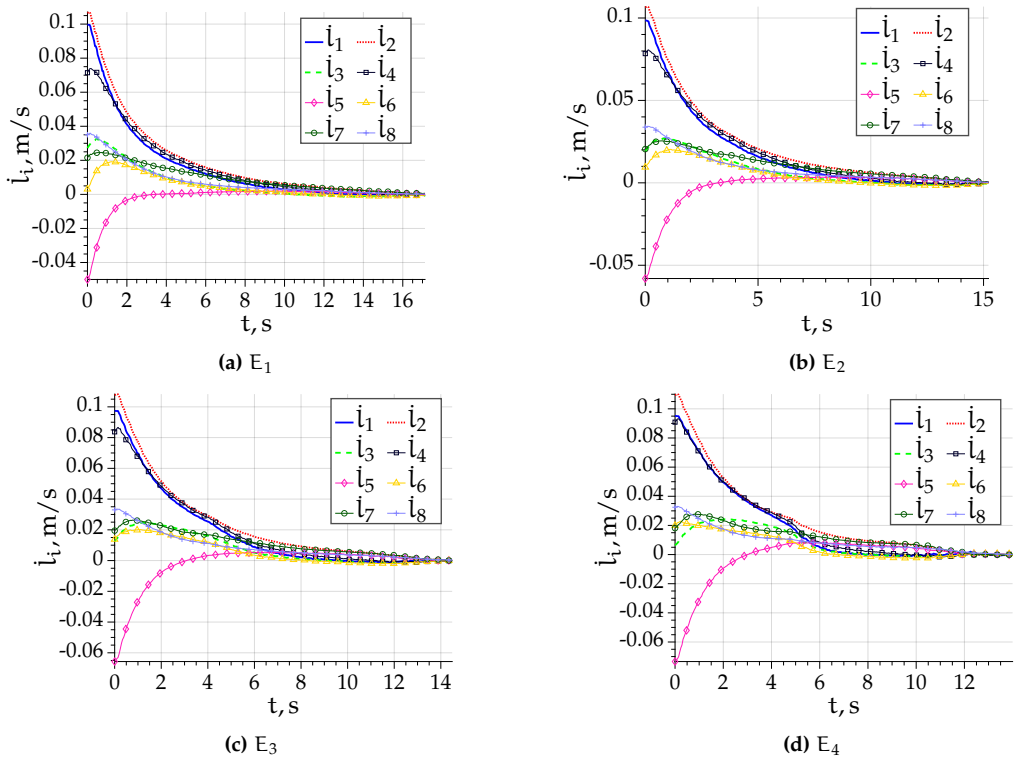
Clearly, perturbations on cable anchor points affect the system more than perturbations on camera position in the moving-platform frame.



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

Figure A.19: Error e over time



(a) E_1 **(b)** E_2

(c) E_3 **(d)** E_4

Figure A.20: Cable velocities over time

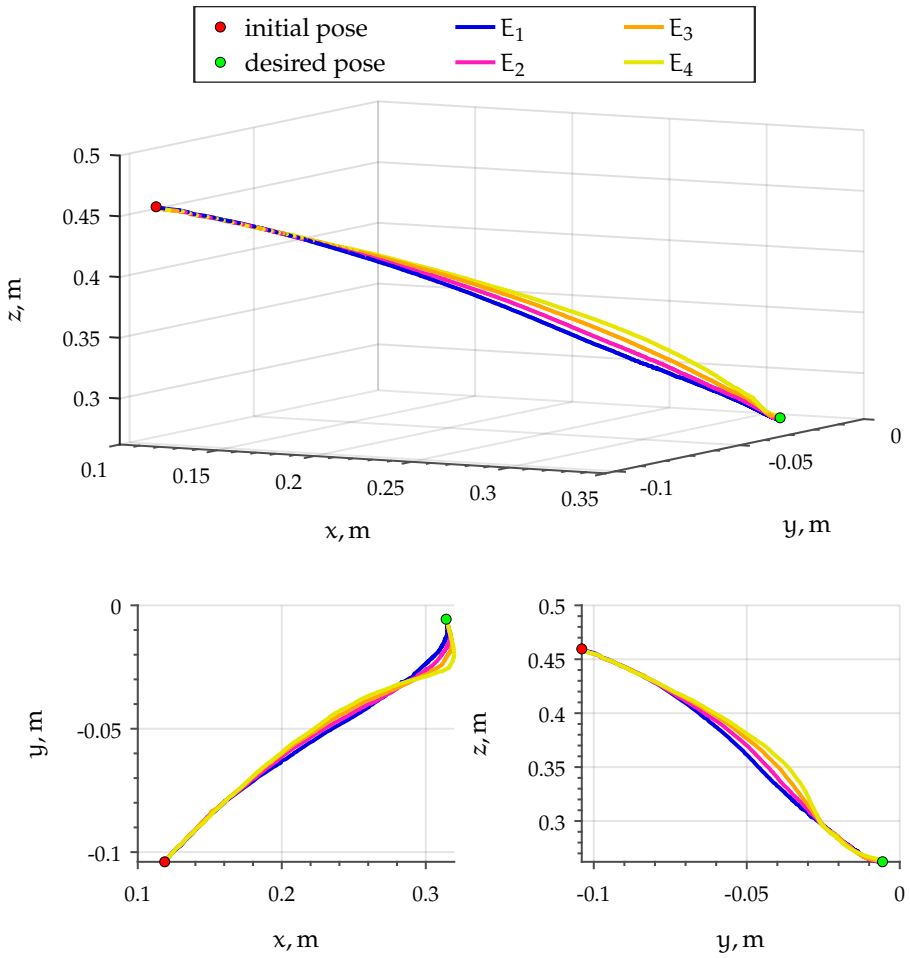


Figure A.21: Moving-platform trajectory

A.6.3 Perturbation on Cable Exit Point Coordinates

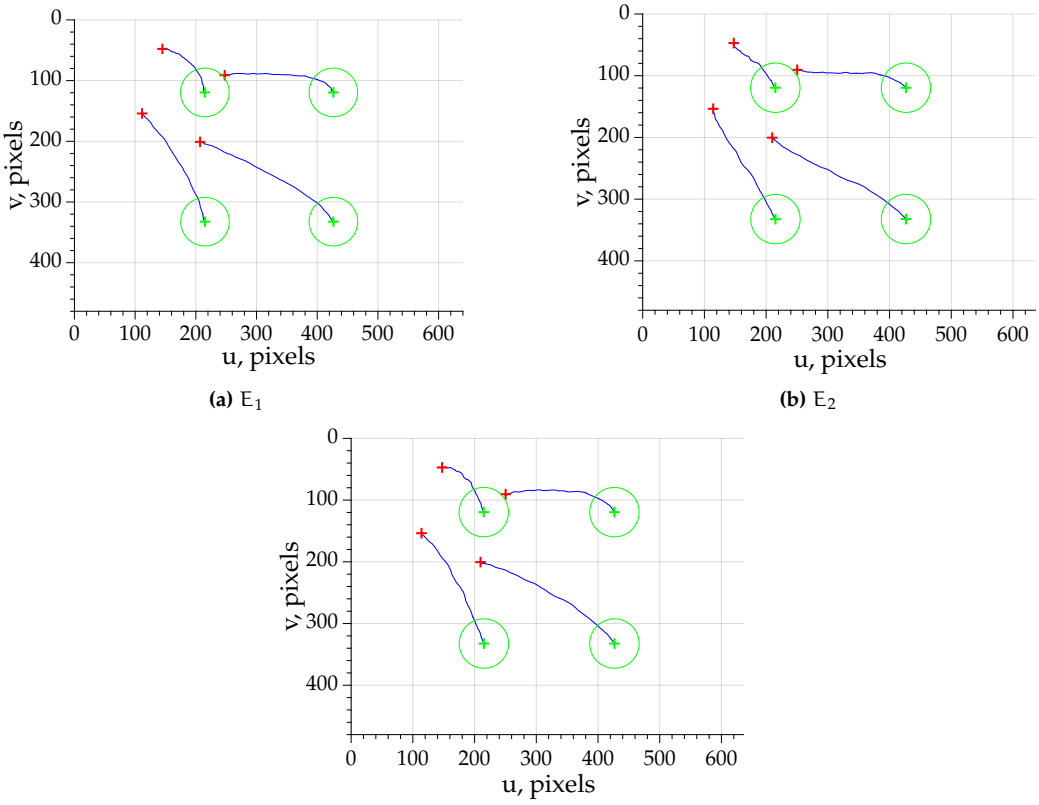
In this section, perturbations are added to cable exit point coordinates. The experiments are defined as follows:

- E_1 - no perturbation added;
- E_2 - $r_{Ai} = 0.032\,\text{m}$;
- E_3 - $r_{Ai} = 0.103\,\text{m}$.

The experimental results are shown in Figs. A.22 through A.25.

In these experiments, the size of the perturbation on cable exit point coordinates reaches that of the previous section on cable anchor point coordinates. On the contrary, the produced trajectories in the image are a lot less perturbed, as can be seen in Fig. A.22. Indeed, the trajectory time is only slightly increased and all the cable velocity plots in Fig. A.24 look identical. Same is true for the moving-platform trajectories shown in Fig. A.25.

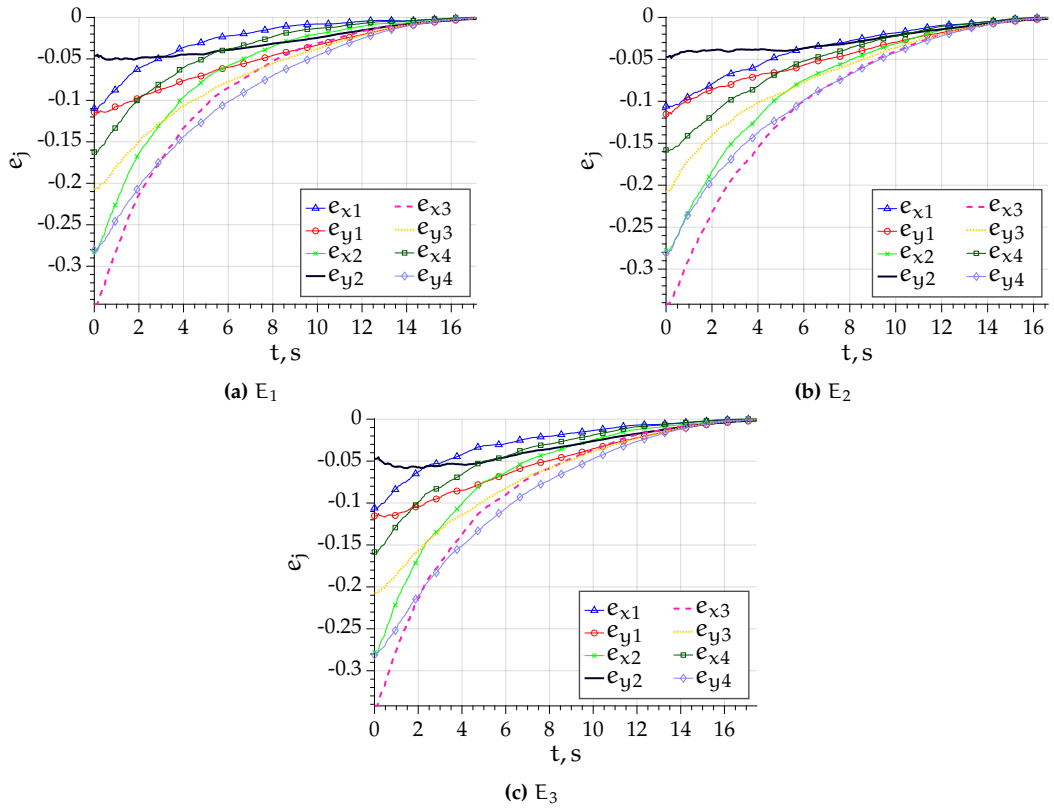
Thus, the system is less sensitive to perturbations on cable exit points than on cable anchor points.



(a) E_1 **(b)** E_2

(c) E_3

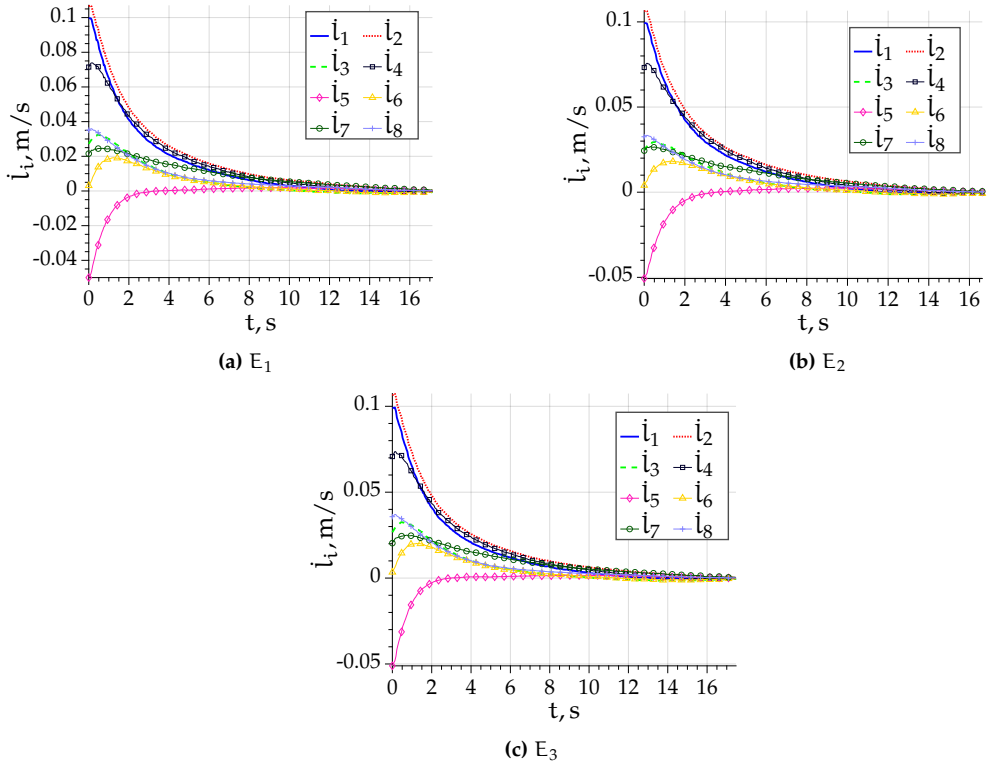
Figure A.22: Trajectory of four points in the image



(a) E_1 **(b)** E_2

(c) E_3

Figure A.23: Error e over time



(a) E_1 **(b)** E_2

(c) E_3

Figure A.24: Cable velocities over time

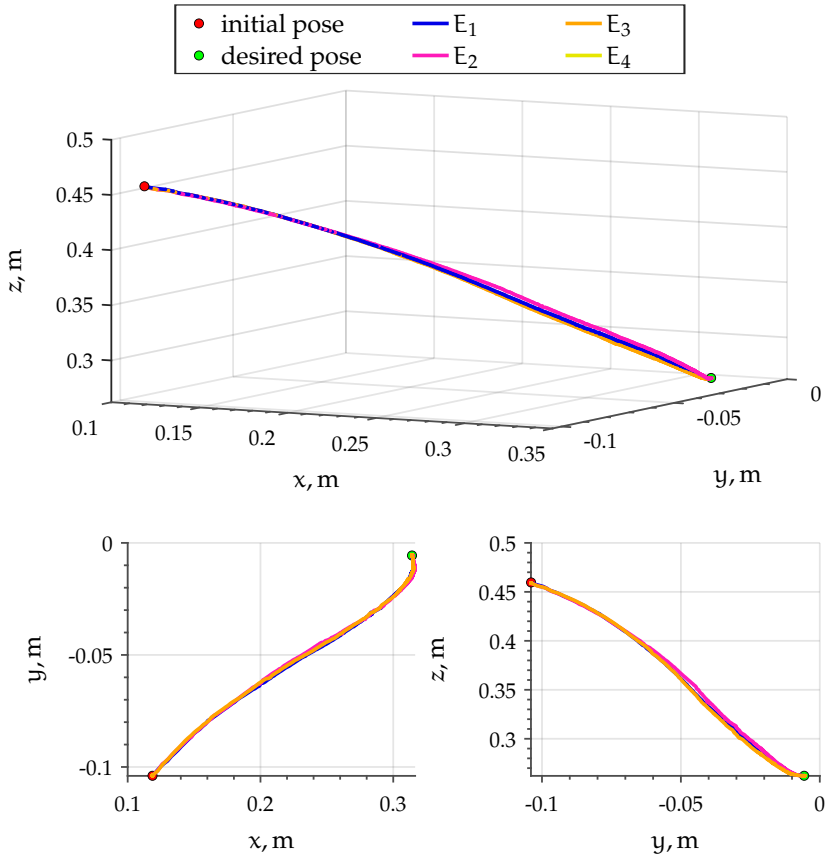


Figure A.25: Moving-platform trajectory

CENTRALE NANTES

Titre : Conception et analyse de stabilité de l'asservissement visuel sur des robots parallèles à câbles pour une amélioration de la précision

Mots clés : Robots à câbles, précision, asservissement visuel, analyse de stabilité, espace de travail, commande

Résumé : Cette thèse présente l'amélioration de la précision des robots parallèles à câbles (RPC) par l'asservissement visuel (AV) et l'utilisation de l'analyse de stabilité pour évaluer la robustesse du système robotique. Les RPC sont une sorte de robots parallèles avec des câbles au lieu de liaisons rigides. Ils sont caractérisés par un grand espace de travail, une charge utile et une reconfigurabilité élevées. En revanche, ils sont généralement peu précis, ce qui les empêche d'être largement utilisés. Avec une caméra embarquée sur la plateforme mobile (PM) et en contrôlant le RPC avec un AV, il est possible d'avoir une grande précision par rapport aux objets qu'elle perçoit. En effet, comme l'objet est constamment observé, la commande ne s'arrête que lorsque la précision souhaitée est atteinte. Cependant, la PM n'est pas observée et sa pose doit être estimée.

Les contributions de cette thèse sont les suivantes. Trois méthodes d'estimation de pose de PM ont été proposées et évaluées. Il a été constaté que l'estimation par intégration de commande est la plus polyvalente. Une analyse de Lyapunov a été réalisée sur un RPC planaire et spatial. Un lien entre la pose de la PM et la stabilité du système a été déterminé et un nouvel espace de travail appelé Control Stability Workspace a été défini. Il a été calculé pour plusieurs approches d'AV sur plusieurs RPC. L'impact de différentes perturbations et erreurs de modélisation a été évalué. Il a été montré que la précision du RPC reste toujours la même tant que le système est stable. Les perturbations du système n'affectent que la trajectoire vers l'objet, qui peut être amélioré en utilisant un suivi de trajectoire. Enfin, pour traiter les pertes de tension des câbles, un algorithme de correction de tension pour l'AV a été proposé et validé.

Title : Design and stability analysis of visual servoing on cable-driven parallel robots for accuracy improvement

Keywords : Cable robots, accuracy, visual servoing, stability analysis, workspace, control

Abstract : This thesis presents accuracy improvement of Cable-Driven Parallel Robots (CDPRs) by visual servoing (VS) and the use of stability analysis to evaluate the robustness of the robotic system. CDPRs are a kind of parallel robots with cables instead of rigid links. They are characterized by a large workspace, a large payload capacity and reconfigurability, including a changeable moving-platform. However, CDPRs lack accuracy, which prevents them to be widely used. With an onboard camera on the moving-platform (MP) used in VS control of CDPRs, it is possible to have high accuracy with respect to a target object. Indeed, as the object is perceived, the control is only stopped when the desired accuracy is achieved. However, the MP is not observed and its pose must be estimated.
The contributions of this thesis are the following. Three moving-platform pose estimation methods were proposed and evaluated on different tasks. It was found that estimation by control integration is the most versatile. Thorough Lyapunov stability analysis was performed on a planar and a spatial CDPRs. A link between the MP pose and system stability was determined and thus a novel workspace named Control Stability Workspace was defined. It was computed for several VS approaches on multiple CDPRs. The impact of different perturbations and modeling errors was evaluated. In experimental validation it was shown that CDPR accuracy always remains the same as long as the system is stable. Perturbations in the system affect only the trajectory to the goal. It was shown that trajectory tracking greatly improves CDPR behavior despite the perturbations. Finally, to deal with cable slackness, a Tension Correction Algorithm for VS was proposed and validated.