
# Warsaw University of Technology

**Faculty of Power and Aeronautical Engineering**

Division of Theory of Machines and Robots



---

Diploma Thesis

Master Degree


Alessia Vignolo


# Visual servoing of the Monash Epicyclic-Parallel Manipulator


Student ID: 263869

Robotics and Control

European Master on Advanced Robotics (EMARO)

**Supervisors:**

Dr. S. Briot (IRCCyN)

Prof. P. Martinet (IRCCyN)

Dr. C. Chen (Monash University)

Dr. P. Malczyk (WUT, Faculty of PAE)

Prof. G. Cannata (Unige)


Warsaw, September 2014

# Statement of the Author of the Thesis
## *(Oświadczenie autora pracy)*

Being aware of my legal responsibility, I certify that this diploma:
*(Świadom odpowiedzialnożci prawnej oświadczam, że przedstawiona praca dyplomowa:)*

- has been written by me alone and does not contain any content obtained in a manner inconsistent with the applicable rules,

- *(zostanapisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obwiązującymi przepisami,)*

- has not been previously subject to the procedures for obtaining professional title or degree at a university.

- *(nie bywcześniej przedmiotem procedur związanych z uzyskaniem tytuzawodowego lub stopnia naukowego w wyższej uczelni.)*

Furthermore I declare that this version of the diploma thesis is identical with the electronic version attached.
*(Oświadczam podnadto, że niniejsza wersja pracy jest identycna z załącznoną wersjią elektroniczną.)*

| ..... | ..... |
|---|---|
| date | author's signature |
| *data* | *podpis autora (autorów) pracy* |

# Visual servoing of the Monash Epicyclic-Parallel Manipulator

Author: Alessia Vignolo

## Abstract

Past research works have proven that the robot end-effector pose of parallel mechanisms can be effectively estimated by vision. For parallel robots, it was previously proposed to directly observed the end-effector. However, this observation may be not possible (e.g. if the robot is milling). Therefore, it has been proposed to use another type of controller based on the observation of the leg directions. Despite interesting results, this controller involves the presence of mapping singularities inside the robot worskpace (near which the accuracy is poor) and it is not suitable for some particular Parallel Kinematics Machine (PKM) families (e.g. MEPaM). This thesis presents a new approach for vision-based control of the end-effector: by observing the mechanism legs, it is possible to extract the Plücker coordinates of their lines and determine the end-effector's pose. A comparison between the previous approach based on the leg direction and the new approach based on the leg line Plücker coordinates is presented. Both are applied on a five-bar mechanism, and it is shown that the new approach has some advantages regarding the workspace of applicability. Further, the new controller is applied to the Monash Epicyclic-Parallel Manipulator (MEPaM) with the results of the simulation presented.

## Streszczenie

W ostatnich badaniach pokazano, że położenie i orientację efektora robota równoległego można efektywnie estymować za pomocą wizji. Dla niektórych typów robotów (np. skrawających) taka estymacja może być trudna do zrealizowania. W związku z tym w piśmiennictwie zaproponowano nowe algorytmy regulacji służące obserwacji członów robota – kończyn. Mimo, że wstępnie uzyskano obiecujące wyniki, wspomniane algorytmy sterujące nie są szczególnie odpowiednie w zastosowaniach dla niektórych grup robotów równoległych (np. MEPaM), ze względu na osobliwości kinematyczne manipulatora wewnątrz przestrzeni roboczej. W tej pracy zaprezentowano nowe podejście do sterowania robotów równoległych bazujące na informacji wizyjnej. Położenie i orientację efektora manipulatora estymuje się poprzez obserwacje kończyn robota za pomocą współrzędnych i algebry Plückera. W pracy zamieszczono porównanie proponowanego rozwiązania na tle istniejących w piśmiennictwie i zamieszczono niektóre wyniki obliczeń symulacyjnych dla pięcioboku przegubowego. Otrzymane rezultaty potwierdzają zalety metody, w szczególności jeśli chodzi o rozmiar przestrzeni roboczej, w której może poruszasię robot. Proponowaną strategię regulacji zaimplementowano dla robota MEPaM (Monash Epicyclic-Parallel Manipulator) i przedstawiono wyniki obliczeń symulacyjnych.

# Contents

# Chapter 1

# Introduction

## 1.1 Motivations and general objectives

Compared to serial robots, parallel kinematic manipulators [1] are stiffer and can reach higher speeds and accelerations [2]. However, their control is troublesome because of the complex mechanical structure, highly coupled joint motions and many other factors (e.g. clearances, assembly errors, etc.) which degrade stability and accuracy.

Many research papers focus on the control of parallel mechanisms (see [3] for a long list of references). It is possible to bypass the complex kinematic structure of the robot and to apply a form of control which uses an external sensor to estimate the pose of the end-effector, reducing the stability and accuracy degradation mentioned earlier.

A proven approach for estimating the end-effector pose is through the use of vision. The most common approach consists of the direct observation of the end-effector pose [4, 5, 6]. In some cases, however, it may prove difficult to observe the end-effector of the robot, e.g. in the case of a machine-tool. A substitute target for the observation must then be chosen while effective candidates for this are the legs of the robot, which are usually designed with slim and rectilinear rods [3].

An application of this technique was performed in [7] where vision was used to derive a visual servoing scheme based on the observation of the legs of a Gough-Stewart (GS) parallel robot [8]. In that method, the leg directions (each direction represented by a 3D unit vector) were chosen as visual primitives and control was derived based on their reconstruction from the image. The approach was applied to several types of robots, such as the Adept Quattro and other robots of the same family [9, 10].

However, it was proven later that:

- The mapping between the leg direction space and the end-effector pose space is not free of singularity which considerably affect the performance in terms of accuracy and do not appear at the same place as the singularity of the controlled robot. Finding the singularity of the mapping is a complicated task which can be considerably simplified by using a tool called “the hidden robot” concept [11]. The hidden robot is a virtual robot whose kinematics represents the mapping between the leg direction space and the end-effector pose space. Thus, the mapping singularities appear if and only if the virtual hidden robot encounters kinematic singularities. A general methodology to find the hidden robot model of any parallel robot controlled by leg-observation-based visual servoing approach has been defined in [11] and

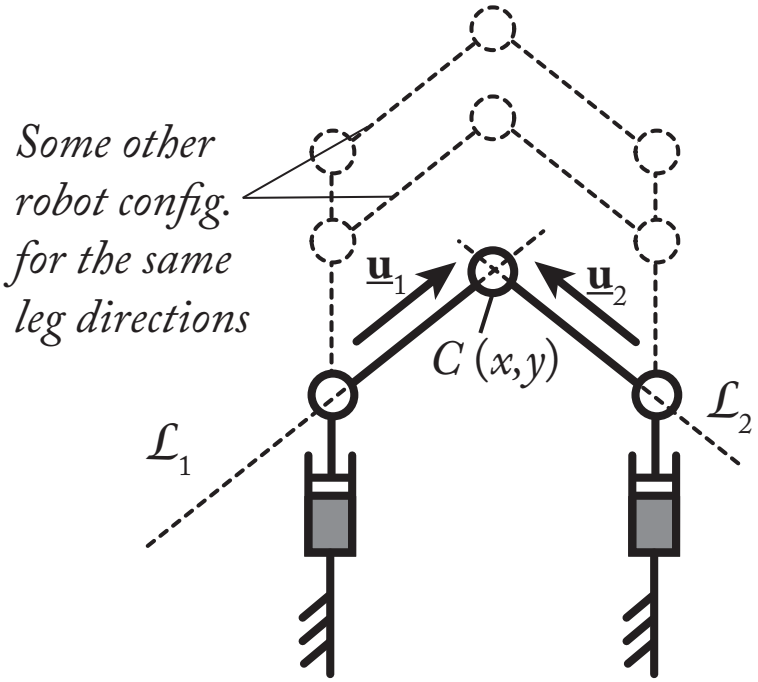



Figure 1.1: The $\underline{P}RRR\underline{P}$ robot

several families of robots have been studied in [12, 13, 11].

- The approach proposed in [7] cannot be applied to any type of robot family: it was shown in [10] that it was not possible to control a particular family of parallel robots for which the first joints of the legs are prismatic joints whose directions are all parallel. For example, in the case of the $\underline{P}RRR\underline{P}$[1] robot with parallel $\underline{P}$ joints (Fig. 1.1), the pose of the end-effector can not be estimated using the leg directions $\underline{\mathbf{u}}_i$ as, for the same values of the vectors $\underline{\mathbf{u}}_1$ and $\underline{\mathbf{u}}_2$, infinite possible configurations of the end-effector can be found.

Regarding this second point, a solution to bypass the mentioned problem would be to use the Plücker coordinates of the lines passing through the legs instead of the leg directions only. Using the Plücker coordinates of the lines passing through the legs for the visual servoing is equivalent to use the leg direction plus their distance and position with respect to the camera frame. Thus, the line passing through the legs are fully defined. Estimating the end-effector pose in the case of the $\underline{P}RRR\underline{P}$ robot of Fig. 1.1 is similar to finding the intersection point of the lines $\mathcal{L}_1$ and $\mathcal{L}_2$ passing through the legs.

The aim of this thesis is dual:

1. First, to introduce this new leg servoing scheme based on the use of the Plücker coordinates of the lines passing through the legs; to apply it on a five-bar mechanism and on MEPaM; and to analyze the singularity of the mapping involved between the observed line space and the end-effector space, and

2. Then, to compare this approach with the previous one based on the leg direction space in

[1] In the following of the text, $R$ and $P$ stand for passive revolute and prismatic joints, respectively, while $\underline{R}$ and $\underline{P}$ stand for active revolute and prismatic joints, respectively.

terms of robustness to measurement noise.

Summarizing, this thesis has the following objectives:

- to develop a simulator of a five-bar mechanism and MEPaM
- to propose a visual-based controller for the five-bar mechanism and MEPaM, based on the observation of the legs (which is the main aim of the thesis)
- to analyze their corresponding hidden robot model.

## 1.2 Software tools

Due to the fact that I do not have access to the prototype at Monash, the research is simulation-based. The tools that have been relevant to my project are:

- Adams, the multibody dynamics and motion analysis software owned by MSC Software Corporation
- Matlab/Simulink, a graphical programming language tool for modeling, simulating and analyzing multidomain dynamic systems developed by MathWorks.

Matlab/Simulink and Adams have been interfaced by Adams/Controls. Within Adams/Controls, it is possible to model a system and then, via some TCP/IP communication as feedback, the model can be exported in Matlab/Simulink. Then, a controller can be built in Simulink: the inputs for the controller are the outputs from the Adams model and the outputs from the controller are the inputs for the Adams model. This closed loop between Simulink and Adams/Controls makes simulation of non linear time-invariant systems very simple.

## 1.3 Organization of the text

The rest of the thesis is organized as follows. Chapter 2 describes the different control approaches which have been used until now and it explains why the Cartesian space control is a valid alternative to the usual joint space control in some machines. Then, there is a summary about the novel approach based on the leg direction developed by the IRCCyN Robotics Team and a conclusion which explains also that this novel approach is not enough for some particular robots, like MEPaM.
Chapter 3 gives a general view on the Plücker coordinates concept and a detailed explanation about their application in the visual servoing field, in particular about how they are used in the new control approach implemented during the thesis work.
Chapter 4 describes both the old leg-direction-based and the new line-based control approaches applied on a five-bar mechanism, analyzing also the singularities introduced by the two controllers. Then, a comparative analysis between them is done and the results obtained using the two approaches are presented, pointing out the advantages of the new controller. Chapter 5 introduces the MEPaM (Monash epicyclic-parallel manipulator), with a description of its design and its kinematics (based both on the actuators angles and the Plücker coordinates of the legs). The new control approach is then applied on it, and an analysis of the real robot and of the hidden robot is also done. Then, the results obtained from the simulation are presented.
Chapter 6 presents the conclusions and the future work.

# Chapter 2

# State of the art about the different methods of control

In this section, there will be a literature review about the existing control strategies, with their major drawbacks pointed out.

## 2.1 Joint space control

In [14] it has been shown that the control designs taken for serial are usually employed in parallel robots, in spite of the unsuitability.
The main control strategies used until now, for both serial and parallel mechanisms, are in the ***joint space***: the linear single-axis and the computed torque control.

The **linear single-axis control**, or PID control, is the most popular and simplest way of control. It is shown in Fig. 2.1, and the symbols have the following meanings:
$X_d$: desired end-effector pose; $IKM$: Inverse kinematic model; $q$ and $q_d$: joint positions and desired joint positions; $e$: error between $q$ and $q_d$, $M$: inertia matrix of the actuated bodies, mapped into the active joint space, diagonal and constant; $s$: Laplace variable; $u_{PID}$: torque input generated by the PID controller; $u_{ff}$: feedforward compensation term; $\Gamma$: actuation torques. The block $s^2$ is computed on-line.

This type of control system gives good performances for a wide range of serial manipulators, but it shows weakness regarding fast serial manipulators, where it leads to a poor dynamic accuracy [15]. In the case of parallel robots, it leads to a poor dynamic accuracy too, for the complex inverse kinematics.
The simplest way for compensating the non-linear dynamic behaviour is the **computed torque control**, which is shown in Fig. 2.2. Here we have some additional symbols, which are: $IDM$: Inverse dynamic model; $\dot{q}$: joint velocities.

In such a way, the system is linearized and, if the dynamic model represents perfectly the real machine [15], the linear controller can work with the maximal performance all over the workspace. Nevertheless, the condition can be verified almost never.
Therefore, joint space control is very good for serial kinematic machines because, for them, the

joint space control is a state feedback control, which is known as the one that can provide the best accuracy. On the other hand, parallel machines state is represented by the end-effector pose: then, the joint space control is not a state feedback control, therefore the best accuracy cannot be achieved with such a control.
Moreover, a joint space control has to include the forward kinematic model in order to find the end-effector pose, but it is not so easy like in serial manipulator. In fact, it is difficult to obtain closed-form solutions form of forward kinematics of parallel robots, and a joint configuration can lead to various end-effector poses.
Since there are a lot of disadvantages in using joint space control and parallel machines are defined by the end-effector pose, a good idea is the ***Cartesian space control***.

## 2.2 Cartesian space control

While for serial mechanism the inverse differential kinematic model depends only on the joint values, for parallel machines it depends also on the end-effector pose. Then, the aim is to drive the error signal, i.e. the difference between the current and the desired end-effector pose, to zero. So, in the Cartesian control, the instantaneous Cartesian velocities which drive the error to zero are computed at each sample time. Therefore, estimating the end-effector pose is needed. In the Fig.2.3, the Cartesian space equivalent of the linear single-axis control generally used is shown. However, it is shown that the transposition from joint to Cartesian space is not completely straightforward. In the control scheme of Fig.2.3, the simplified dynamics is expressed as

$$\gamma = \hat{D^T}(X) I \ddot{X} \tag{2.1}$$

where only the inertia of the end-effector $I$ is taken into account. The latter is mapped into the active joint space with the forward instantaneous kinematic matrix.

## 2.3 End-effector pose estimation

Estimating the end-effector pose can be done through joint measurements and the forward kinematic model, but it is difficult due to the lack of an analytic formulation of the forward kinematic model of a parallel mechanism. The inverse kinematic model is, instead, analytically defined for most of the parallel mechanisms and one could numerically invert it. However, the numerical inversion implies high order polynomial root determination with several solutions (up to 40 real solutions for a GS platform).
Therefore, generally the solution of this problem is not unique, i.e. given actuated joint coordinates correspond to several ways of assembling a parallel manipulator, and it is usually impossible to express analytically the generalized coordinates as functions of the actuated joint coordinates.
The methods for finding all of the solutions of the direct kinematic problem have a computational time which, although decreasing, are still too high for using them in real time control. Here is a summary of these methods [16]:

- **elimination**: this method allows to reduce the initial problem of solving a system of several equations to a problem of solving a univariate polynomial equation. This method requires algebraic equations (i.e polynomial equations) for the inverse kinematics, and it has two main drawbacks: it is very difficult to get a final polynomial which has the minimal degree, and

the calculation of its coefficients is very sensitive to numerical errors (therefore this method does not provide certified solutions).

- **homotopy**: this method requires algebraic equations for the inverse kinematics too, but it may provide certified solutions. The main drawback is its slowness.
- **Gröbner basis**: this method requires algebraic equations for the inverse kinematics too, and, if the coefficients of the inverse kinematics equations are rational numbers, provides certified solutions.
- **interval analysis**: this method can be used even if the inverse kinematics equations are not algebraic, and provides certified solutions. The unknowns must be bounded (but this is usually the case for kinematics problems). The drawback is that the computation time is difficult to estimate.
- **ad-hoc methods**: the direct kinematics problem is transformed into a different and simpler problem that is solved with one of the above method or with an optimization procedure. These methods can be fast, but are thought for a specific architecture, and cannot easily be extended to other architectures.

These methods above are for obtaining *all* the solutions of the direct kinematic, but the aim of the real direct kinematic problem is to determine the *current* pose of the end-effector, so the pose when the joint values were measured. The problem is that there is no known algorithm which can select the right current pose among the set of solutions. Another problem is the certification of the solutions: in some of the methods above, the calculation may imply a high number of operations, being very sensitive to numerical errors. In this case it is necessary to check the validity of the solutions. Another problem is that it has to be assumed that all the data used to establish the inverse kinematic equations are exact: this assumption is usually false, because the modeling of the robot does not exactly fit the real robot and the joint variables are uncertain. The last problem is that all of them are too slow for real-time use, so they cannot be used for control purposes.
There are some methods which can calculate the current pose, but all of them need a priori information of the pose. They are numerical methods and are based on the concept that the unknown current pose should be close to the pose established when the direct kinematic problem was solved the previous time. Therefore, the solution of a system starts with an estimate close to the current solution. One of the most classical numerical method for solving a non-linear system of equations is the **Newton iterative scheme**. This method is more compatible with a real time context, but has still some drawbacks. One problem is that Newton scheme may not to converge. Moreover, there can be convergence problem if the inverted Jacobian matrix is close to be singular. Therefore this method may give an incorrect answer or no answer at all, leading to a wrong control.
Fortunately, it is possible to detect if the Newton method has converged to an incorrect solution: knowing the sample time of the controller, the maximal velocity of the end-effector and its last pose, one can derive the extremal values which the new pose parameters cannot exceed. It is not possible to detect a wrong solution just in case more than one solution satisfies the extremal values constraints, but it usually does not occur in a slow moving robot, in fact the closeness of two solutions implies the inverse of the Jacobian nearly singular, leading to a failure of the calculation of it.
For control purposes, the best alternative is to use a direct kinematics procedure that may not be the fastest available in absolute term, but the one that leads to the most reliable result, provided

that the additional computation time, compared to the fastest method, does not exceed the sampling time.
For real time context, the numerical methods seem to be more suitable. Nevertheless, as said before, these numerical methods use the a-priori information on the current pose, but in some cases this is not available (e.g. when starting the robot), therefore it is important to find a method for which it is not required.

On the other hand, one of the most promising alternatives is the **"'metrological redundancy"'**, which adds sensors in the mechanism for simplifying the kinematic models and the control.
Computer vision can be use in an efficient way for estimating the end-effector pose and, then, for Cartesian control in parallel mechanisms. There are different possible approaches:
1) **Vision as a sensor**, where the end-effector pose is estimated by vision and then transformed into joint configuration through the inverse kinematic model. Then the control is done in the joint space. Computer vision can be considered as a contact-less redundant sensor, so the simplified model based on the redundant metrology can be used. However, the joint control does not take into account the kinematic closures, yielding high internal forces.
2) **Visual servoing**, where the measure of the end-effector pose is directly used for control. In this case, the Cartesian desired velocity is generated and then converted through the inverse Jacobian into joint actuation. In the parallel case is rather easier than in the serial case, since the inverse Jacobian of a parallel mechanism has a straightforward expression.
There are more techniques like position-based visual servoing (PBVS) [17] (explicit pose measurement) to image-based visual servoing (IBVS) [18] (it is made implicitly by using only image measurements). The control loop is closed on the vision sensor, and this leads to a high robustness to calibration and perturbations errors. Indeed, these errors only appear in a Jacobian matrix but not in the regulated error.
However, these two ways of using vision are efficient but not innovative. Moreover, the direct application of visual servoing techniques assumes implicitly that the robot inverse differential kinematic model is given and that it is calibrated. Therefore, modeling, identification and control have small interaction with each other. Indeed, the model is usually defined for control using proprioceptive sensing only and does not foresee the use of vision for control, because identification and control are defined later with the constraints imposed by the model. This is useful for modularity but this might not be efficient in terms of accuracy as well as of experimental set-up time.
A new approach was proposed [19] in order to take the advantages of redundant metrology and visual servoing, and it will be explained in subsection 2.3.1.

### 2.3.1 Novel approach of visual servoing

The **novel approach** proposed in [19] solves most of the previous techniques difficulties seen in section 2.3. For instance, adding redundant sensors may be impossible or, anyway, you have to modify the mechanism to install them. Moreover, observing the end-effector is not always possible: the case of MEPaM is explicative because, since it is a haptic device, the presence of the user hand may be an obstacle to do it. However, usually the observation of the legs is not a problem: this turns the vision from an exteroceptive to a proprioceptive sensor.
This approach, based on the observation of the legs with a camera fixed with respect to the base,

has been validated and illustrated on different parallel mechanism. The first step was made on the GS platform (Fig. 2.4) simulation [19]: the visual primitives were represented by the leg directions, and their reconstruction from the image was used for deriving the control. Then this approach was improved in terms of accuracy by using the observation of the leg edges and controlling the robot directly in the image space [20]: this new method was applied to Adept Quattro robot (Fig. 2.5) and other machines of its family [21] [22].

In the novel approach, it is taken into account that control will be performed using vision since the first stages of modeling: for that reason, the fusion between robot kinematic and projective geometry is needed (Fig. 2.6). The point of unification can be found in line geometry, because of the shape of parallel mechanism legs: they are usually slim and rectilinear, therefore they can be modeled as straight lines.

### 2.3.2 Hidden robot

Recently, it has been shown that the visual servoing of the leg directions of the GS platform and the Adept Quattro with 3 translational DOFs is equivalent to controlling other virtual robots, with assembly modes and singular configurations different from those of the real ones, which can be considered "'hidden"' within the controller.
It has been demonstrated in [23] that the concept of hidden robot:

1. can be used to explain why the observed robot with $n$ legs can be controlled using the observation of only $m$ leg directions ($m < n$) arbitrarily chosen, and can also help to choose the best set of legs to observe with respect to some given performance indices,

2. can be used to prove that is not always a full diffeomorphism between the Cartesian space and the leg direction space, and can also bring solutions for avoiding the convergence to a non desired pose,

3. allows to simplify the analysis of singularities of the mapping between the leg direction space and the Cartesian space by reducing the problem to the analysis of singularities of a new robot,

4. can be used to certify that the robot will not converge to local minima.

Usually, the control method is based on the measurement of the actuators motions, while the novel approach described in section 2.3.1 is based on the observation of the leg directions. Therefore, it is interesting to understand what type of virtual actuators this observation corresponds to. For instance, it is possible to answer to this question taking into account the Gough-Stewart case, which has six $U\underline{P}S$ legs of varying length $q_i, i \in 1...6$, (U, P and S stand for passive universal, prismatic and spherical joints, while $\underline{U}$, $\underline{P}$ and $\underline{S}$ stand for active universal, prismatic and spherical joints) attached to the base by $U$ joints located in points $\mathbf{A}_i$ and to the platform by $S$ joints located in points $B_i$ (Fig. 2.4, Right). By analyzing the leg $i$, the unit vector ${}^b\underline{\mathbf{u}}_i$ can be parameterized by two independent coordinates, such as the angles defined by the universal ($U$) joint rotations: ${}^b\underline{\mathbf{u}}_i$ is a measure of $U$ joint displacements, so the observation corresponds to the virtual actuator of the $U$ joint. Therefore, observing the directions of the leg is useful to control the displacement of a virtual $\underline{U}PS$ leg instead of a $U\underline{P}S$ leg. It is known that a $3-\underline{U}PS$ robot (Fig. 2.7) is fully actuated: because of this, observing just three legs instead of six is enough to control the GS

platform. Therefore, if only three legs are observed, the visual servoing of the leg directions of the GS platform is equivalent to controlling another robot, the $3-U\underline{P}S$, which has the same geometric properties of the GS robot (same attachment points, same leg lengths, same $U$ and $S$ joints orientations), but with different assembly modes and singular configurations.
In order to get a correct control, they should be analyzed through a forward geometric problem. Considering the $3-\underline{U}PS$ robot of the Fig. 2.7, it is possible to note that, disassembling the leg 3 at point $\mathbf{B}_3$, only four actuators remain to control the six mobilities, and the end-effector gains two DOFs. This gained motion is named *Cardanic motion*, and it is defined by the points $\mathbf{B}_1$ and $\mathbf{B}_2$ which are constrained on the lines of ${}^b\underline{\mathbf{u}}_1$ and ${}^b\underline{\mathbf{u}}_2$ and by the platform which can rotate around the line $\mathbf{B}_1\mathbf{B}_2$. The surface described by the point $\mathbf{B}_3$ is an octic surface, which is an algebraic surface of degree eight. The point $\mathbf{B}_3$ is, instead, constrained to move on the line of ${}^b\underline{\mathbf{u}}_3$. As demonstrated in [11], a line and an octic surface can have up to eight real intersection points, therefore the assembly modes of the $3-U\underline{P}S$ robot can be up to eight.
These different assembly modes explain why the end-effector does not always converge to the desired position, even if the observed directions do.
It has also been shown that the legs to observe and the number of them should be chosen carefully in order to avoid inaccuracy problems.
Regarding the number of the legs to observe, in [23] it was shown that in the case of GS platform, the pose accuracy of the robot can be improved adding measurement redundancy, therefore by observing four, five or six legs instead of the minimum required, which is three legs. However, increasing the number of legs to observe leads to an increase of the computational time and it is an important drawback when high sampling rates are required. Thus, a compromise must be found between the sampling period and the computational time for any given application.
Regarding the selection of the legs, it is necessary to ensure that:

- the legs have to be observable during all the time of the robot displacement
- the initial and final robot configurations must be included in the same assembly mode of the virtual $3-\underline{U}PS$ robot. Otherwise, the controller is not be able to converge to the desired end-effector pose, although the observed leg directions do.

Moreover, in order to achieve the best possible final accuracy, the following procedure can be used:

1. knowing the six leg orientations at the initial and final GS platform configurations, compute the solutions of the forward geometric model of the different possible $3-\underline{U}PS$ robots,
2. find the solutions of the forward geometric model that belong to the same assembly modes; if, for one given virtual robot, initial and final platform configurations do not belong to the same assembly mode, discard it;
3. for all remaining virtual $3-\underline{U}PS$ robots, knowing the observation error, compute the positioning error; hold the set of legs that guaranty the best accuracy;
4. test the controller (in simulation) with the held set of legs; if there is no problem of convergence and the legs are observable during the whole displacement, the problem is solved; if not, discard this set of leg and redo point c; if it does not exist any $3-\underline{U}PS$ robot for which initial and final configurations belong to the same assembly mode, the displacement is not feasible.

Therefore, the concept of hidden robot model is an important tool useful to analyze the intrinsic properties of some controllers developed by the visual servoing community.
In order to apply the novel approach to a parallel robot, the analysis of its hidden robot is needed.

## 2.4 Conclusions

Through the state of the art, an overview on the control systems was done, showing the lacks of the joint space control systems and the advantages of the Cartesian space control systems in the case of parallel kinematic machines. Then, several methods for finding all the possible solutions and the current one of the direct kinematic problem in order to estimate the pose of the end-effector were reviewed, pointing out some drawbacks.
After, the novel approach proposed by some researchers of the IRCCyN Robotics Team was analyzed. The method, based on the observation of the legs with a camera fixed with respect to the base, seems to be suitable to control the haptic device MEPaM. Nevertheless, this novel approach has some problems which have to be solved.
In fact, for applying the vision-based control, it is necessary to ensure a diffeomorphism between the error $\mathbf{e}$, which has to be regulated, and the Cartesian state $\mathbf{x}$ of the end effector, which means that controlling $\mathbf{e}$ becomes equivalent to controlling $\mathbf{x}$. Therefore, as $\mathbf{e}$ depends only on the direction of the legs, this approach can not be applied to some PKM families. One problem occurs when it is not possible to observe the state of mechanism: for instance, you can consider the case in which there is no change of direction of the legs but the end-effector moves, like in the case of MEPaM where the directions of the cylindrical legs are constant. Another problem is about internal motions, because the leg directions do not represent it.
These problems are due to the fact that this control approach is based just on a unit vector for describing the leg direction, and can be solved using a visual feature more relevant for those particular PKM families: the Plücker coordinates.

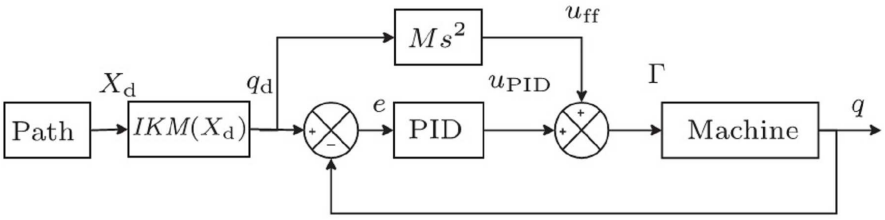



Figure 2.1: Single axis control.

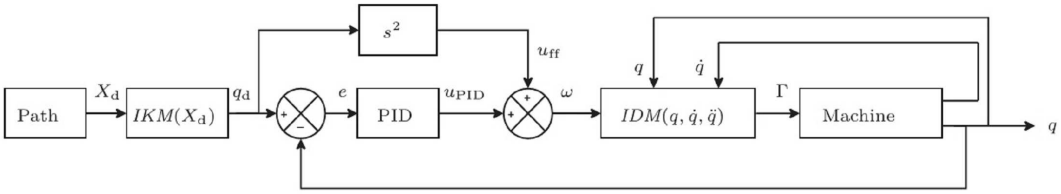



Figure 2.2: Computed torque control

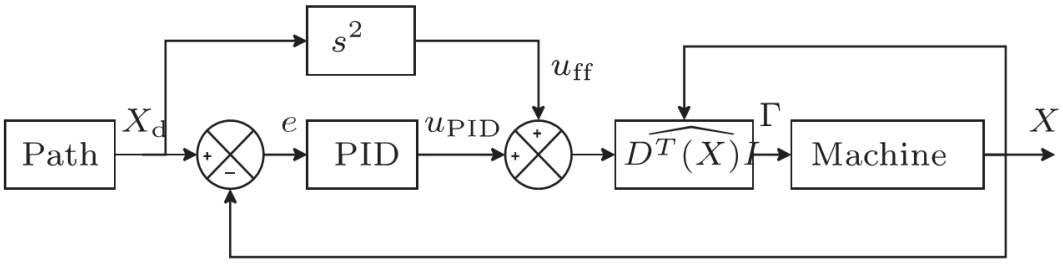



Figure 2.3: Simple PID control in theCartesian space

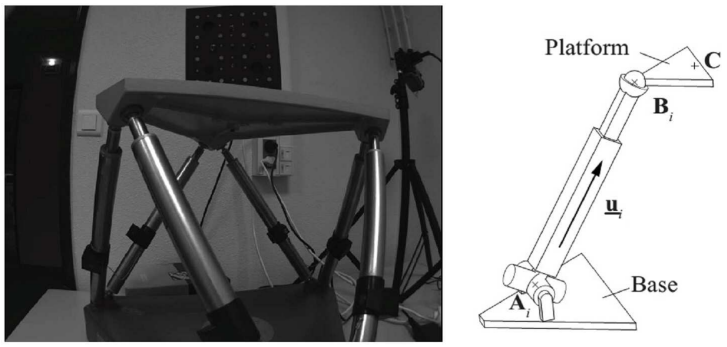



Figure 2.4: Left: a Gough-Stewart platform from DeltaLab; Right: schematics of leg $i$

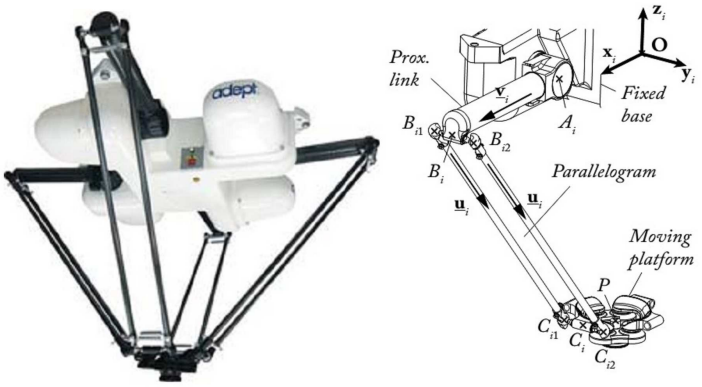



Figure 2.5: Left: Adept Quattro robot; Right: schematics of leg $i$

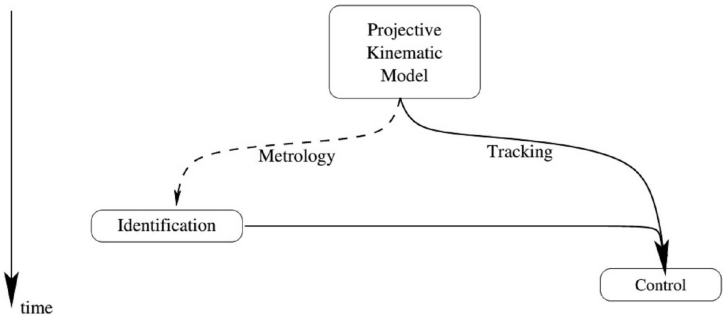



Figure 2.6: Simplified cascade from modeling to vision-based control using a projective kinematic model.

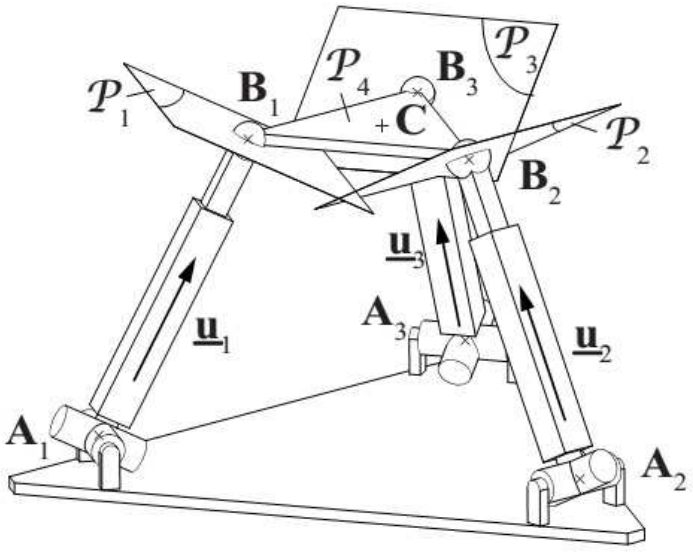


Figure 2.7: Schematics of a $3-U\underline{P}S$ robot.

# Chapter 3

# Plücker coordinates for visual servoing

## 3.1 General definition and geometric meaning

Plücker coordinates are a representation of lines in $\mathbb{R}^3$. It is known that a line $\mathcal{L}$ in 3-dimensional Euclidean space can be determined by two distinct points that it contains. Let the points $\mathbf{x} = (x_1, x_2, x_3)$ and $\mathbf{y} = (y_1, y_2, y_3)$ be the points contained in the line $\mathcal{L}$, then the vector displacement from $\mathbf{x}$ to $\mathbf{y}$ represents the direction of the line. That is, every displacement between points on $\mathcal{L}$ is a scalar multiple of $\mathbf{d} = \mathbf{y} - \mathbf{x}$. Suppose that a physical particle of unit mass moves from $\mathbf{x}$ to $\mathbf{y}$, it would have a moment about the origin. The geometric equivalent is a vector with the direction perpendicular to the plane containing $\mathcal{L}$ and the origin, and with the length equals the double of the area of the triangle formed by the segment of the displacement and the origin. Therefore, the moment is the vector cross product $\mathbf{m} = \mathbf{x} \times \mathbf{y}$. The area of the triangle is proportional to the length of the segment between $\mathbf{x}$ and $\mathbf{y}$. By definition, the moment vector is perpendicular to each displacement along the line, so the vector dot product is $\mathbf{d} \cdot \mathbf{m} = 0$.
The two values $\mathbf{d}$ and $\mathbf{m}$ are sufficient to uniquely determine $\mathcal{L}$. Therefore, the Plücker coordinates are given by $(\mathbf{d} : \mathbf{m}) = (d_1 : d_2 : d_3 : m_1 : m_2 : m_3)$.
Furthermore, this approach can be extended to points, lines, and planes.

## 3.2 Leg observation

Both control schemes, the leg-direction-based and the line-based ones, are based on the fact that it is possible to observe by observing the robot legs. In this Section, the way to extract the leg direction and the Plücker coordinates of the line passing through the leg is discussed.

### 3.2.1 Line modeling

A line $\mathcal{L}$ in space, expressed in the camera frame, is defined by its Binormalized Plücker coordinates [24]:

$$\mathcal{L} \equiv ({}^c\underline{\mathbf{u}}, {}^c\underline{\mathbf{n}}, {}^c n) \tag{3.1}$$

where ${}^c\underline{\mathbf{u}}$ is the unit vector giving the spatial orientation of the line[1], ${}^c\underline{\mathbf{n}}$ is the unit vector defining the so-called interpretation plane of line $\mathcal{L}$ and ${}^cn$ is a nonnegative scalar. The latter are defined by ${}^cn{}^c\underline{\mathbf{n}} = {}^c\mathbf{P} \times {}^c\underline{\mathbf{u}}$ where ${}^c\mathbf{P}$ is the position of any point $P$ on the line, expressed in the camera frame. Notice that, using this notation, the well-known (normalized) Plücker coordinates [25, 2] are the couple $({}^c\underline{\mathbf{u}}, {}^cn{}^c\underline{\mathbf{n}})$.

The projection of such a line in the image plane, expressed in the camera frame, has for characteristic equation [24]:

$$ {}^c\underline{\mathbf{n}}^T {}^c\mathbf{p} = 0 \tag{3.2}$$

where ${}^c\mathbf{p}$ are the coordinates in the camera frame of a point $P$ in the image plane, lying on the line.

With the intrinsic parameters $\mathbf{K}$, one can obtain the line equation in pixel coordinates ${}^p\underline{\mathbf{n}}$ from:

$$ {}^p\underline{\mathbf{n}}^T {}^p\mathbf{p} = 0 \tag{3.3}$$

Indeed, replacing ${}^p\mathbf{p}$ with $\mathbf{K}{}^c\mathbf{p}$ in this expression yields:

$$ {}^p\underline{\mathbf{n}}^T \mathbf{K} {}^c\mathbf{p} = 0 \tag{3.4}$$

By identification of (3.2) and (3.3), one obtains

$$ {}^p\underline{\mathbf{n}} = \frac{\mathbf{K}^{-T} {}^c\underline{\mathbf{n}}}{\|\mathbf{K}^{-T} {}^c\underline{\mathbf{n}}\|}, {}^c\underline{\mathbf{n}} = \frac{\mathbf{K}^T {}^p\underline{\mathbf{n}}}{\|\mathbf{K}^T {}^p\underline{\mathbf{n}}\|} \tag{3.5}$$

Notice that for numerical reasons, one should use normalized pixel coordinates. Namely, let us define the pixel frame by its origin located at the image center (i.e. the intersection of the image diagonals) and such that the pixel coordinates vary approximately between $-1$ and $+1$, according to the choice of the normalizing factor, which can be the image horizontal dimension in pixels, or its vertical dimension, or its diagonal.

### 3.2.2 Cylindrical leg observation

The legs of parallel robots have usually cylindrical cross-sections [2]. The edges of the $i$-th cylindrical leg are given, in the camera frame, by [26] (Fig. 3.2):

$$ {}^c\underline{\mathbf{n}}_i^1 = -\cos\theta_i {}^c\underline{\mathbf{h}}_i - \sin\theta_i {}^c\underline{\mathbf{u}}_i \times {}^c\underline{\mathbf{h}}_i \tag{3.6}$$

$$ {}^c\underline{\mathbf{n}}_i^2 = +\cos\theta_i {}^c\underline{\mathbf{h}}_i - \sin\theta_i {}^c\underline{\mathbf{u}}_i \times {}^c\underline{\mathbf{h}}_i \tag{3.7}$$

where $\cos\theta_i = \sqrt{{}^ch_i^2 - R_i^2} \big/ {}^ch_i$, $\sin\theta_i = R_i / {}^ch_i$ and $({}^c\underline{\mathbf{u}}_i, {}^c\underline{\mathbf{h}}_i, {}^ch_i)$ are the Binormalized Plücker coordinates of the cylinder axis and $R_i$ is the cylinder radius.

It was also shown in [26] that the leg orientation, expressed in the camera frame, is given by

$$ {}^c\underline{\mathbf{u}}_i = \frac{{}^c\underline{\mathbf{n}}_i^1 \times {}^c\underline{\mathbf{n}}_i^2}{\left\| {}^c\underline{\mathbf{n}}_i^1 \times {}^c\underline{\mathbf{n}}_i^2 \right\|} \tag{3.8}$$

Let us remark now that each cylinder edge is a line in space, with Binormalized Plücker expressed in the camera frame $({}^c\underline{\mathbf{u}}_i, {}^c\underline{\mathbf{n}}_i^j, {}^cn_i^j)$ (Fig 3.2(a)). Moreover, any point $A_i$ (of coordinates ${}^c\mathbf{A}_i$ in the camera frame) lying on the cylinder axis is at the distance $R_i$ from the edge. Consequently,

[1] In the following of the thesis report, the superscript before the vector denotes the frame in which the vector is expressed ("$b$" for the base frame, "$c$" for the camera frame and "$p$" for the prixel frame). If there is no superscript, the vector can be written in any frame.

a cylinder edge is entirely defined by the following constraints, expressed here in the camera frame, although valid in any frame:

$$ {}^c\underline{\mathbf{n}}_i^{jT}\,{}^c\mathbf{A}_i = -R_i \tag{3.9} $$

$$ {}^c\underline{\mathbf{n}}_i^{jT}\,{}^c\underline{\mathbf{n}}_i^{j} = 1 \tag{3.10} $$

$$ {}^c\underline{\mathbf{u}}_i^{T}\,{}^c\underline{\mathbf{n}}_i^{j} = 0 \tag{3.11} $$

The vector ${}^c\mathbf{h}_i = {}^ch_i\,{}^c\underline{\mathbf{h}}_i$ can be computed using the edges of the $i$-th cylindrical leg too, and it is given by

$$ {}^c\mathbf{h}_i = {}^c\mathbf{D}_i \times {}^c\underline{\mathbf{u}}_i \tag{3.12} $$

where ${}^c\mathbf{D}_i$ is the position of the point $B_i$ in the camera frame, which is the closest point of the axis of the $i$-th leg to the camera. It is given by

$$ {}^c\mathbf{D}_i = \frac{R_i}{\sin(\theta_i)} \cdot \frac{{}^c\mathbf{n}_i^1 + {}^c\mathbf{n}_i^2}{\left\| {}^c\mathbf{n}_i^1 + {}^c\mathbf{n}_i^2 \right\|} \tag{3.13} $$

## 3.3 Pinhole camera

The model that has been used for the visual servoing is a pinhole camera because it is simple to implement and is a good approximation of real cameras. In the Fig. 3.3 a camera with $O$ as center of projection and the principal axis parallel to $Z$ axis is shown. The distance between $O$ and the image plane is the focal length $f$. The 3D point $P = (X, Y, Z)$ is projected on the image plane at coordinates $P_c = (u, v)$. $P_c$ can be found using the properties of similar triangles as

$$ \frac{f}{Z} = \frac{u}{X} = \frac{v}{Y} \tag{3.14} $$

which gives

$$ u = \frac{fX}{Z} \tag{3.15} $$

$$ v = \frac{fY}{Z} \tag{3.16} $$

If the origin of the 2D image coordinate system does not coincide with the intersection point between the Z axis and the image plane, $P_c$ needs to be translated to the desired origin. With a translation of $(t_u, t_v)$ we have

$$ u = \frac{fX}{Z} + t_u \tag{3.17} $$

$$ v = \frac{fY}{Z} + t_v \tag{3.18} $$

Using homogenous coordinates for $P_c$ it becomes

$$ \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & t_u \\ 0 & f & t_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.19} $$

Indeed it gives $P_c = (u, v, w) = (\frac{fX}{Z}, \frac{fY}{Z}, 1)$. In Eq. 3.19, $P_c$ is expressed in meters, and, for converting it in pixel, we need the resolution of the camera in pixels/meters. Assuming that the pixels are square, the resolution would be equal in both $u$ and $v$ directions of the camera image coordinates. For generality, we assume rectangle pixels with resolutions $m_u$ and $m_v$ pixels/meters in $u$ and $v$ direction respectively. Therefore, to have $P_c$ in pixels, we should multiply $u$ and $v$ by $m_u$ and $m_v$ respectively, and Eqs. 3.17 and 3.18 become

$$u = m_u \frac{fX}{Z} + m_u t_u \tag{3.20}$$

$$v = m_v \frac{fY}{Z} + m_v t_v \tag{3.21}$$

and can be expressed in matrix form as

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} m_u f & 0 & m_u t_u \\ 0 & m_v f & m_v t_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \alpha_x & 0 & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} P = KP \tag{3.22}$$

K is called the *intrinsic parameter* matrix of the camera because it depends on its intrinsic parameters, such as the focal length and the principal axis. In the case where the image coordinate axes $u$ and $v$ are not orthogonal to each other, $K$ also has a skew parameter $s$, and it is given by

$$K = \begin{bmatrix} \alpha_x & s & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.23}$$

If the camera does not have its center of projection in $(0, 0, 0)$ and its oriented in an arbitrary way, then a rotation and a translation are needed to make the camera coordinate system coincide with the XYZ coordinate system of Fig. 3.3. Let $T$ be the camera translation and $R$ the camera rotation, then the matrix obtained by first applying the translation and then the rotation is given by the $3 \times 4$ matrix $(R|RT)$ that is called the *extrinsic parameter* matrix. Therefore, the projection of $P$ is given by

$$P_c = K(R|RT)P = CP \tag{3.24}$$

where C is a $3 \times 4$ matrix called complete camera calibration matrix. $P$ need to be in 4D homogeneous coordinates and $P_c$ is obtained in 3D homogenous coordinates and in pixels. The 2D location of the projection on the camera plane will be obtained by dividing the first two coordinates of $P_c$ by the third one.

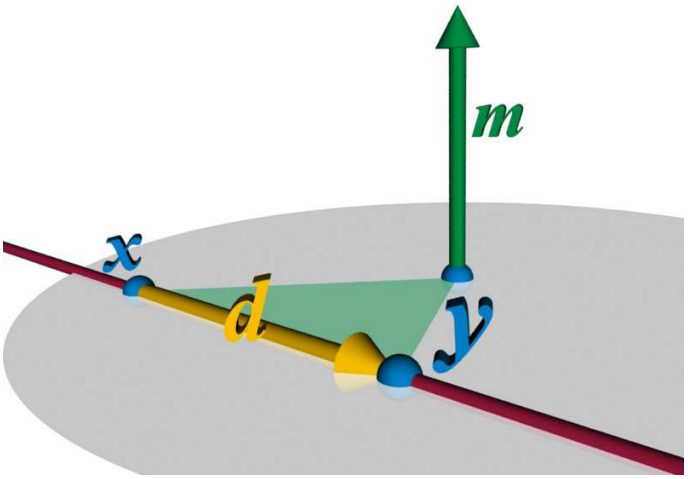



Figure 3.1: Geometric intuition of the Plücker coordinates of a line. **x** and **y** are two points on the line, whose displacement is **d** and whose moment is **m**.

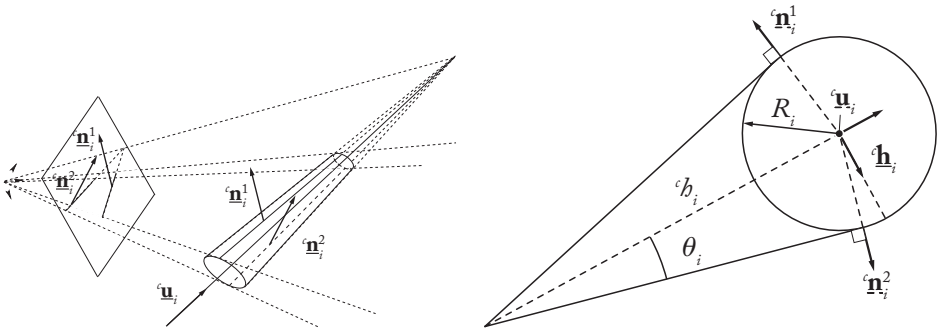



Figure 3.2: Projection of a cylinder in the image (a) and its visual edges (b)

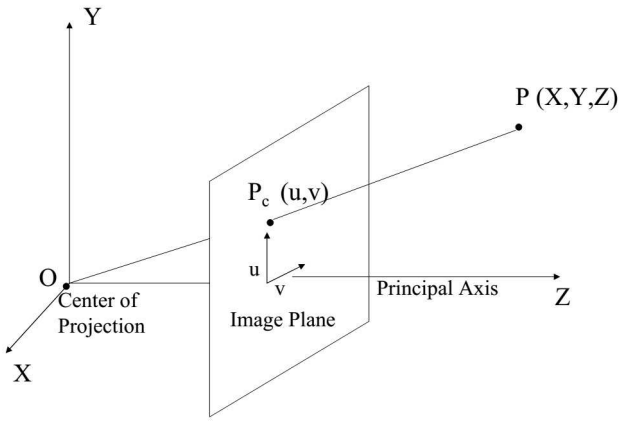



Figure 3.3: A pinhole camera model

# Chapter 4

# Leg-direction-based and line-based visual servoing of the five-bar mechanism

## 4.1 Visual servoing schemes

In this Section, the control schemes for the visual servoing of the five-bar mechanism are defined and compared. Vectors are represented in bold letters, unit vectors in underlined and bold letters, matrices in capital and bold letters.

### 4.1.1 Kinematics of the five-bar mechanism

The planar five-bar mechanism (Fig. 4.1) is a 2 degrees-of-freedom (*dof*) parallel robot able to achieve two translations in the plane $(O, \mathbf{x}_0, \mathbf{y}_0)$ and which is composed of two legs:

- A leg composed of 3 $R$ joints with an axis directed along $\mathbf{z}_0$ and located at points $A_1$, $B_1$ and $C$, the joint located at point $A_1$ being actuated, and
- A leg composed of 2 $R$ joints with an axis directed along $\mathbf{z}_0$ and located at points $A_2$ and $B_2$, the joint located at point $A_2$ being actuated,

all other joints being passive. Thus, the vector of actuated coordinates is $\mathbf{q}^T = [q_1\ q_2]$. The end-effector is located at point $C$ and its controlled coordinates along $\mathbf{x}_0$ and $\mathbf{y}_0$ are denoted as $x$ and $y$, respectively.

The position of the point $C$ is given by, for $i = 1, 2$:

$$\mathbf{C} = \mathbf{A}_i + l_{1i}\underline{\mathbf{v}}_i + l_{2i}\underline{\mathbf{u}}_i \tag{4.1}$$

in which:

- $\mathbf{C}$ is the position of the point $C$, while $\mathbf{A}_i = [\delta_i\ 0]^T$ ($\delta_1 = -l_{OA_i}$ and $\delta_2 = +l_{OA_i}$) is the position of the point $A_i$ (the frame is not specified, but it is usually either the base frame or the camera frame),
- $l_{1i}$ and $l_{2i}$ denote the length of the links $A_iB_i$ and $B_iC$ respectively,

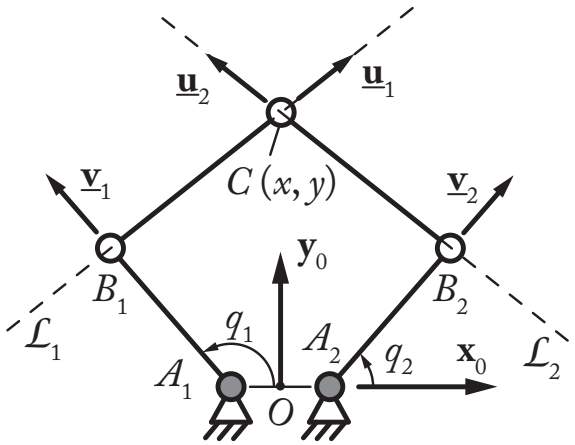



Figure 4.1: The planar five-bar mechanism (the gray pairs denote the actuated joints)

- vectors $\underline{\mathbf{v}}_i$ and $\underline{\mathbf{u}}_i$ are unit vectors defining the direction of the links $A_iB_i$ and $B_iC$ respectively.

Rearranging (4.1), we obtain

$$\mathbf{C} - \mathbf{A}_i - l_{1i}\underline{\mathbf{v}}_i = l_{2i}\underline{\mathbf{u}}_i \tag{4.2}$$

Then, squaring both sides of (4.2) we get, for $i = 1, 2$:

(x-$\delta_i - l_{1i}\cos q_i)^2 + (y - l_{1i}\sin q_i)^2 = l_{2i}^2$
$x^2 + \delta_i^2 + l_{1i}^2\cos^2 q_i - 2x\delta_i - 2xl_{1i}\cos q_i + 2\delta_i l_{1i}\cos q_i + y^2 + l_{1i}^2\sin^2 q_i - 2yl_{1i}\sin q_i = l_{2i}^2$
$x^2 + \delta_i^2 + l_{1i}^2 - 2x\delta_i - 2xl_{1i}\cos q_i + 2\delta_i l_{1i}\cos q_i + y^2 - 2yl_{1i}\sin q_i = l_{2i}^2$
$2l_{1i}(\delta_i - x)\cos q_i - 2yl_{1i}\sin q_i + x^2 + \delta_i^2 - 2x\delta_i + l_{1i}^2 + y^2 - l_{2i}^2 = 0$

Naming $a_i = 2l_{1i}(\delta_i - x)$, $b_i = -2yl_{1i}$, $c_i = (x-\delta_i)^2 + y^2 + l_{1i}^2 - l_{2i}^2$ and substituting $\cos q_i = \frac{1-t^2}{1+t^2}$ and $\sin q_i = \frac{2t}{1+t^2}$, where $t = \mathrm{tg}(\frac{q_i}{2})$, we get

$$a_i\frac{1-t^2}{1+t^2} + b_i\frac{2t}{1+t^2} + c_i = 0 \tag{4.3}$$

$$t^2(c_i - a_i) + 2b_i t + a_i + c_i = 0 \tag{4.4}$$

and it comes that:

$$q_i = 2\,\mathrm{tg}^{-1}\left(\frac{-b_i \pm \sqrt{b_i^2 - c_i^2 + a_i^2}}{c_i - a_i}\right) \tag{4.5}$$

The first-order kinematics that relates the platform translational velocity $\boldsymbol{\tau}_p$ to the actuator velocities can be obtained through the differentiation of (4.2) with respect to time and leads to

$$2(x - \delta_i - l_{1i}\cos q_i)\dot{x} + 2(y - l_{1i}\sin q_i)\dot{y} + 2(l_{1i}\sin q_i(x - \delta_i) - yl_{1i}\cos q_i)\dot{q}_i = 0 \tag{4.6}$$

Putting Eq. 4.6 for $i = 1, 2$ in matrix form, we have

$$\underbrace{2\begin{bmatrix} x-\delta_1-l_{11}\cos q_1 & y-l_{11}\sin q_1 & 0 \\ x-\delta_2-l_{12}\cos q_1 & y-l_{12}\sin q_2 & 0 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix}}_{\boldsymbol{\tau}_p} + \underbrace{2\begin{bmatrix} l_{11}\sin q_1(x-\delta_1)-yl_{11}\cos q_1 & 0 \\ 0 & l_{12}\sin q_2(x-\delta_2)-yl_{12}\cos q_2 \end{bmatrix}}_{\mathbf{B}} \underbrace{\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}}_{\dot{\mathbf{q}}} = 0 \tag{4.7}$$

Thus,

$$\mathbf{A}_{[2x3]}\boldsymbol{\tau}_{p[3x1]} + \mathbf{B}_{[2x2]}\dot{\mathbf{q}}_{[2x1]} = \mathbf{0} \tag{4.8}$$

$$\boldsymbol{\tau}_p = -\mathbf{A}^{-1}\mathbf{B}\dot{\mathbf{q}} = \mathbf{J}\dot{\mathbf{q}} \tag{4.9}$$

or also

$$\dot{\mathbf{q}} = -\mathbf{B}^{-1}\mathbf{A}\boldsymbol{\tau}_p = \mathbf{J}^{+}\boldsymbol{\tau}_p \tag{4.10}$$

### 4.1.2 Leg-direction-based visual servoing of the five-bar mechanism

**Kinematics of the five-bar mechanism using the leg-direction-based visual servoing technique**

The control of the five-bar mechanism using the leg-direction-based visual servoing technique developed in [7] proposes to observe the distal leg direction $\underline{\mathbf{u}}_i$ to control the robot displacements. $\underline{\mathbf{u}}_i$ can be obtained directly from (4.2)

$$\underline{\mathbf{u}}_i = (\mathbf{C} - \mathbf{A}_i - l_{1i}\underline{\mathbf{v}}_i)/l_{2i} \tag{4.11}$$

Differentiating (4.11) with respect to time leads to:

$$\dot{\underline{\mathbf{u}}}_i = \left(\boldsymbol{\tau}_p - l_{1i}\frac{d\underline{\mathbf{v}}_i}{dt}\right)/l_{2i} \tag{4.12}$$

Knowing that $\frac{d\underline{\mathbf{v}}_i}{dt} = \frac{d}{dt}\begin{bmatrix}\cos q_i \\ \sin q_i \\ 0\end{bmatrix}\dot{q}_i = \begin{bmatrix}-\sin q_i \\ \cos q_i \\ 0\end{bmatrix}\dot{q}_i = \underline{\mathbf{v}}_i^{\perp}\,\dot{q}_i$, Eq. 4.12 becomes

$$\dot{\underline{\mathbf{u}}}_i = \left(\boldsymbol{\tau}_p - l_{1i}\underline{\mathbf{v}}_i^{\perp}\,\dot{q}_i\right)/l_{2i} \tag{4.13}$$

Finally, from (4.10), we have that $\dot{q}_i = -\mathbf{a}_i/b_{ii}\boldsymbol{\tau}_p$, where $\mathbf{a}_i$ is the $i$-th row of $\mathbf{A}$ and $b_{ii}$ is the $i$-th diagonal element of $\mathbf{B}$, and Eq. 4.13 becomes

$$\dot{\underline{\mathbf{u}}}_i = \left(\mathbf{I}_{[3x3]} + l_{1i}\underline{\mathbf{v}}_i^{\perp}\mathbf{a}_i/b_{ii}\right)/l_{2i}\;\boldsymbol{\tau}_p = \mathbf{M}_{ui}^{T}\;\boldsymbol{\tau}_p \tag{4.14}$$

where $\mathbf{I}_{[3x3]}$ is the $(3\times3)$ identity matrix and matrix $\mathbf{M}_{ui}^{T}$ is called the interaction matrix related to the $i$-th leg.

It can be proven that matrix $\mathbf{M}_{ui}^{T}$ is of rank 1. As a result, a minimum of two independent legs is necessary to control the end-effector pose. An interaction matrix $\mathbf{M}_{u}^{T}$ can then be obtained by stacking the matrices $\mathbf{M}_{ui}^{T}$ of the two legs ($i = 1, 2$). Suming the two legs, we have

$$\dot{\mathbf{u}}_{[6x1]} = \mathbf{M}_{u[6x3]}^{T}\boldsymbol{\tau}_{p[3x1]} \tag{4.15}$$

where $\mathbf{u} = \begin{bmatrix}\underline{\mathbf{u}}_1 \\ \underline{\mathbf{u}}_2\end{bmatrix}$.

**Control scheme and interaction matrix**

Visual servoing is based on the so-called interaction matrix $\mathbf{M}^T$ [27] which relates the velocity of the end-effector $\boldsymbol{\tau}_p$ to the time derivative of the vector $\mathbf{s}$, which is the vector of the visual primitives that are used through:

$$\dot{\mathbf{s}} = \mathbf{M}^T \boldsymbol{\tau}_p \tag{4.16}$$

The vector $\mathbf{s}$ in the leg-direction-based controller is $\mathbf{s} = \mathbf{u}$ and we call the interaction matrix $\mathbf{M}_u^T$.

The visual primitives vector is formed by unit vectors, therefore it is more elegant to use the geodesic error rather than the standard vector difference. Consequently, the error will be:

$$\mathbf{e}_i = \underline{\mathbf{u}}_i \times \underline{\mathbf{u}}_{di} \tag{4.17}$$

where $\underline{\mathbf{u}}_{di}$ is the desired value of $\underline{\mathbf{u}}_i$.

Finally, a control is chosen such that $\mathbf{e}$, the vector stacking the errors $\mathbf{e}_i$ associated to $k$ legs ($k = 2..4$), decreases exponentially, i.e.

$$\dot{\mathbf{e}} = -\lambda \mathbf{e} \tag{4.18}$$

From Eq. 4.17, we have

$$\dot{\mathbf{e}}_i = -\left[\underline{\mathbf{u}}_{di}\right]_\times \dot{\underline{\mathbf{u}}}_i \tag{4.19}$$

where $[...]_\times$ is the antisymmetric matrix associated to a 3D vector [6]). Therefore, we have

$$\dot{\mathbf{e}} = \begin{bmatrix} \dot{\mathbf{e}}_1 \\ \dot{\mathbf{e}}_2 \end{bmatrix} = -\begin{bmatrix} [\underline{\mathbf{u}}_{d1}]_\times \dot{\underline{\mathbf{u}}}_1 \\ [\underline{\mathbf{u}}_{d2}]_\times \dot{\underline{\mathbf{u}}}_2 \end{bmatrix} = -\underbrace{\begin{bmatrix} [\underline{\mathbf{u}}_{d1}]_\times & \mathbf{0} \\ \mathbf{0} & [\underline{\mathbf{u}}_{d2}]_\times \end{bmatrix}}_{\mathbf{U}} \cdot \underbrace{\begin{bmatrix} \dot{\underline{\mathbf{u}}}_1 \\ \dot{\underline{\mathbf{u}}}_2 \end{bmatrix}}_{\dot{\mathbf{u}}} \tag{4.20}$$

From Eqs. 4.18, 4.1.2 and 4.15 we have

$$-\lambda \mathbf{e}_{[6x1]} = -\mathbf{U}_{[6x6]} \dot{\mathbf{u}}_{[6x1]} = \underbrace{-\mathbf{U}_{[6x6]} \mathbf{M}_{u[6x3]}^T}_{\mathbf{N}_{[6x3]}^T} \boldsymbol{\tau}_{p[3x1]} \tag{4.21}$$

$$\boldsymbol{\tau}_p = \mathbf{N}^{T+} \cdot (-\lambda \mathbf{e}) \tag{4.22}$$

This expression can be transformed into the control joint velocities using Eq. (4.10):

$$\dot{\mathbf{q}} = \mathbf{J}^+ \cdot \boldsymbol{\tau}_p = -\lambda \mathbf{J}^+ \mathbf{N}^{T+} \mathbf{e} \tag{4.23}$$

### 4.1.3 Line-based visual servoing of the five-bar mechanism

In the present subsection, the controller based on the estimation of the Plücker coordinates of the lines passing through the legs is defined. This is the first time that such a controller is proposed.

**Kinematics of the five-bar mechanism using the line-based visual servoing technique**

The control of the five-bar mechanism using the new line-based visual servoing technique proposes to extract the Plücker coordinates $(\underline{\mathbf{u}}_i, \mathbf{h}_i)$ of the two legs attached to the end-effector in order to control the robot displacements. The control can be done thanks to the fact that the point to control $C$ is the intersection point of the lines of the two observed cylindrical legs. Applying the formula of the intersection point between two lines in a plane both expressed in Plücker coordinates, the position of the point $C$ expressed in Plücker coordinates is given by, for $i = 1, 2$ [28]:

$$\mathbf{C}_P = (-(\mathbf{h}_1 \cdot \mathbf{N}) \cdot \underline{\mathbf{u}}_2 + (\mathbf{h}_2 \cdot \mathbf{N}) \cdot \underline{\mathbf{u}}_1 + (\mathbf{h}_1 \cdot \underline{\mathbf{u}}_2) \cdot \mathbf{N} : (\underline{\mathbf{u}}_1 \times \underline{\mathbf{u}}_2) \cdot \mathbf{N}) \tag{4.24}$$

in which:

- $(\underline{\mathbf{u}}_1, \mathbf{h}_1)$ and $(\underline{\mathbf{u}}_2, \mathbf{h}_2)$ are the Plücker coordinates of the 1st and the 2nd leg respectively,
- $\mathbf{N}$ is a unit vector along a coordinate axis, with $(\underline{\mathbf{u}}_1 \times \underline{\mathbf{u}}_2) \cdot \mathbf{N}$ non-zero.

For converting the Plücker coordinates of the point in non-homogeneous coordinates, the formula is

$$\mathbf{C} = \frac{\mathbf{C}_P}{(\underline{\mathbf{u}}_1 \times \underline{\mathbf{u}}_2) \cdot \mathbf{N}}. \tag{4.25}$$

Moving the right term of (4.24) to the left side, extending it and naming the equations with $f_i$ for $i = 1..3$ leads to

$$f_1 = x + \frac{h_{1z}u_{2x} - h_{2z}u_{1x}}{u_{1x}u_{2y} - u_{2x}u_{1y}} = 0 \tag{4.26}$$

$$f_2 = y + \frac{h_{1z}u_{2y} - h_{2z}u_{1y}}{u_{1x}u_{2y} - u_{2x}u_{1y}} = 0 \tag{4.27}$$

$$f_3 = z + \frac{-h_{1x}u_{2x} - h_{1y}u_{2y}}{u_{1x}u_{2y} - u_{2x}u_{1y}} = 0 \tag{4.28}$$

where $\mathbf{C} = (x, y, z, 1)$.

Differentiating (4.26), (4.27), (4.28) with respect to time leads to

$$\begin{aligned}\dot{f}_1 = \dot{x} + \frac{-h_{2z}(u_{1x}u_{2y} - u_{1y}u_{2x}) - (h_{1z}u_{2x} - h_{2z}u_{1x})u_{2y}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{1x} + \frac{(h_{1z}u_{2x} - h_{2z}u_{1x})u_{2x}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{1y} + \\ \frac{h_{1z}(u_{1x}u_{2y} - u_{1y}u_{2x}) + (h_{1z}u_{2x} - h_{2z}u_{1x})u_{1y}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{2x} - \frac{(h_{1z}u_{2x} - h_{2z}u_{1x})u_{1x}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{2y} + \\ \frac{u_{2x}}{u_{1x}u_{2y} - u_{1y}u_{2x}}\dot{h}_{1z} - \frac{u_{1x}}{u_{1x}u_{2y} - u_{1y}u_{2x}}\dot{h}_{2z} = 0\end{aligned} \tag{4.29}$$

$$\begin{aligned}\dot{f}_2 = \dot{y} - \frac{(h_{1z}u_{2y} - h_{2z}u_{1y})u_{2y}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{1x} + \frac{-h_{2z}(u_{1x}u_{2y} - u_{1y}u_{2x}) + (h_{1z}u_{2y} - h_{2z}u_{1y})u_{2x}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{1y} + \\ \frac{u_{1y}(h_{1z}u_{2y} - u_{1y}h_{2z})}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{2x} + \frac{h_{1z}(u_{1x}u_{2y} - u_{1y}u_{2x}) - (h_{1z}u_{2y} - h_{2z}u_{1y})u_{1x}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{2y} + \\ \frac{u_{2y}}{u_{1x}u_{2y} - u_{1y}u_{2x}}\dot{h}_{1z} - \frac{u_{1y}}{u_{1x}u_{2y} - u_{1y}u_{2x}}\dot{h}_{2z} = 0\end{aligned} \tag{4.30}$$

$$\begin{aligned}\dot{f}_3 = \dot{z} - \frac{(-h_{1x}u_{2x} - h_{1y}u_{2y})u_{2y}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{1x} + \frac{(-h_{1x}u_{2x} - h_{1y}u_{2y})u_{2x}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{1y} + \\ \frac{-h_{1x}(u_{1x}u_{2y} - u_{1y}u_{2x}) + (-h_{1x}u_{2x} - h_{1y}u_{2y})u_{1y}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{2x} + \frac{-h_{1y}(u_{1x}u_{2y} - u_{1y}u_{2x}) - (-h_{1x}u_{2x} - h_{1y}u_{2y})u_{1x}}{(u_{1x}u_{2y} - u_{1y}u_{2x})^2}\dot{u}_{2y} + \\ \frac{-u_{2x}}{u_{1x}u_{2y} - u_{1y}u_{2x}}\dot{h}_{1x} - \frac{u_{2y}}{u_{1x}u_{2y} - u_{1y}u_{2x}}\dot{h}_{1y} = 0\end{aligned} \tag{4.31}$$

Finally, putting (4.29), (4.30), (4.31) in matrix form, it comes that

$$\mathbf{I}_{[3x3]}\boldsymbol{\tau}_{p[3x1]} + \mathbf{P}_{[3x12]}\dot{\mathbf{l}}_{[12x1]} = 0 \tag{4.32}$$

$$\dot{\mathbf{l}} = -\mathbf{P}^{+} \cdot \boldsymbol{\tau}_p = \mathbf{M}_l^T \cdot \boldsymbol{\tau}_p \tag{4.33}$$

where $\mathbf{l} = \begin{bmatrix} \underline{\mathbf{u}}_1 \\ \underline{\mathbf{u}}_2 \\ \mathbf{h}_1 \\ \mathbf{h}_2 \end{bmatrix}$ and $p_{jk} = \frac{\partial f_j}{\partial l_k}$ is the term of the $j$-th row and the $k$-th column of $\mathbf{P}$, with $j = 1..3$ and $k = 1..12$.

**Control scheme and interaction matrix**

The vector $\mathbf{s}$ in the line-direction-based controller is $\mathbf{s} = \mathbf{l}$ and we call the interaction matrix $\mathbf{M}_l^T$.

Because the vectors $\mathbf{h}_1$ and $\mathbf{h}_2$ are not unit, the control law (4.16) cannot be used as it. Consequently, it is necessary to use the following error

$$\mathbf{e}_i = \mathbf{l}_i - \mathbf{l}_{di} \tag{4.34}$$

where $\mathbf{l}_{di}$ is the desired value of $\mathbf{l}_i$.
The control is chosen in the same way of (4.18). From (4.18), (4.33) and (4.34), it comes

$$-\lambda\mathbf{e} = \dot{\mathbf{l}} - \dot{\mathbf{l}}_d = \mathbf{M}_l^T \cdot \boldsymbol{\tau}_p - \dot{\mathbf{l}}_d = \mathbf{M}_l^T \cdot \mathbf{J}\dot{\mathbf{q}} - \dot{\mathbf{l}}_d \tag{4.35}$$

Then it is easy to derive the following control joint velocities

$$\dot{\mathbf{q}} = \mathbf{J}^{+} \cdot \mathbf{M}_l^{T+} \cdot (-\lambda\mathbf{e} + \dot{\mathbf{l}}_d) \tag{4.36}$$

where $\mathbf{J}^{+}$ is the inverse Jacobian matrix of the robot which relates the end-effector twist to the actuator velocities, i.e. $\mathbf{J}^{+}\boldsymbol{\tau}_p = \dot{\mathbf{q}}$.

The control scheme is represented in Fig. 4.2.

## 4.2 Analysis of the controller singularities

In this Section, the control schemes for the visual servoing of the five-bar mechanism are compared in terms of singularities.

### 4.2.1 Singularities of the leg-direction-based visual servoing controller

As mentioned in the introduction, the singularity of the mapping involved into the present controller can be analyzed through the aid of the “hidden robot” concept [11].
Indeed, it has been shown that the visual servoing of the leg directions of the GS platform and the Adept Quattro with 3 translational *dof* is equivalent to controlling other virtual robots, with assembly modes and singular configurations different from those of the real ones, which can be considered "‘hidden"’ within the controller.
The robotic community has developed many tools (e.g. the Grassmann algebra) to analyze the singularity of the kinematic architecture of a real mechanism, that is of the usual mapping $\begin{bmatrix} x & y & z \end{bmatrix}^T =$

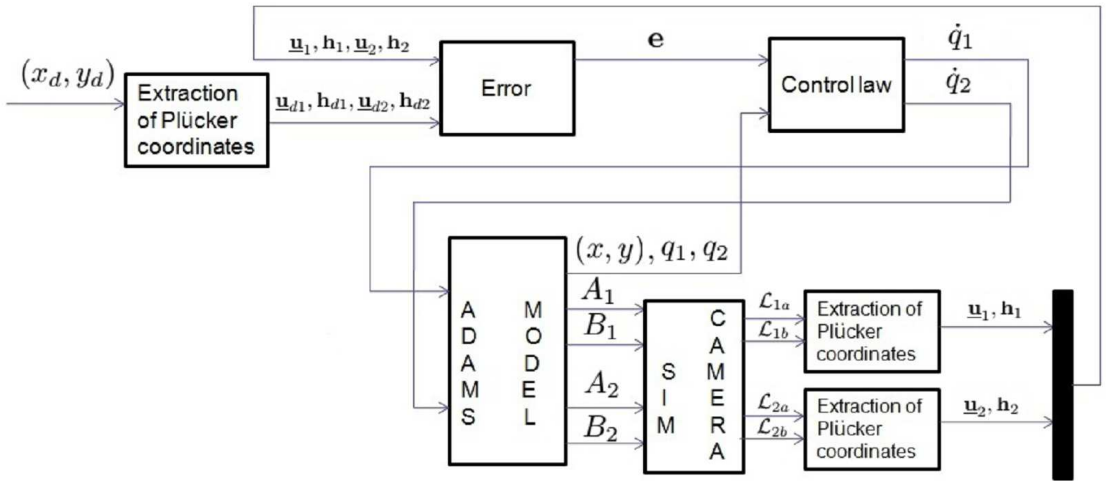



Figure 4.2: Line-based position controller for the five-bar mechanism.
$(x_d, y_d)$: desired position of the end-effector,
$\underline{\mathbf{u}}_1, \mathbf{h}_1, \underline{\mathbf{u}}_2, \mathbf{h}_2$ : Plücker coordinates of the first and the second leg respectively,
$\underline{\mathbf{u}}_{d1}, \mathbf{h}_{d1}, \underline{\mathbf{u}}_{d2}, \mathbf{h}_{d2}$ : desired Plücker coordinates of the first and the second leg respectively,
$\mathbf{e}$ : error vector,
$\dot{q}_1, \dot{q}_2$: joint velocities,
$(x, y)$: end-effector position,
$q_1, q_2$: joint angles,
$A_1, B_1$: two points of the first leg,
$A_2, B_2$: two points of the second leg,
$\mathcal{L}_{1a}, \mathcal{L}_{1b}$: the two visual edges of the first leg,
$\mathcal{L}_{2a}, \mathcal{L}_{2b}$: the two visual edges of the second leg

$f(\mathbf{q})$. In the visual servoing, having to deal with a different map such that $\begin{bmatrix} x & y & z \end{bmatrix}^T = g(\mathbf{s})$, things are more complicated. For simplifying it, it is possible to find a kinematic architecture which can describe the relation $\begin{bmatrix} x & y & z \end{bmatrix}^T = g(\mathbf{s})$, that is an architecture whose actuators $\mathbf{q}$ are related to the visual primitives $\mathbf{s}$: this architecture is the hidden robot. The Jacobian matrix of the hidden robot will be the interaction matrix of the real robot, therefore analyzing the structural singularity of the hidden robot would be the same as analyzing the singularities of the visual servoing controller. The reader willing to have further explanations is referred to [11].

The hidden robot involved into the leg-direction-based visual servoing approach for the five-bar mechansim is shown in Fig. 4.3.

This virtual mechanism is made of two passive planar parallelogram joints $A_iB_iD_iE_i$ linked onto the ground on which is fixed an actuator at point $B_i$ controlling the direction of the link $B_iC$. This special arrangement of the leg makes it possible, for one given position of the actuator at $B_i$, to maintain the orientation with respect to the base of the link $B_iC$ independently of the configuration of the passive parallelogram joint.

A simple kinematic analysis of this virtual robot shows that:

- The Type 1 (or serial) singularities [29] appear when one leg is fully stretched or folded, such as for the five-bar mechansim (Fig. 4.4),
- The Type 2 (or parallel) singularities [29] appear when the links $A_1B_1$ and $A_2B_2$ are parallel,

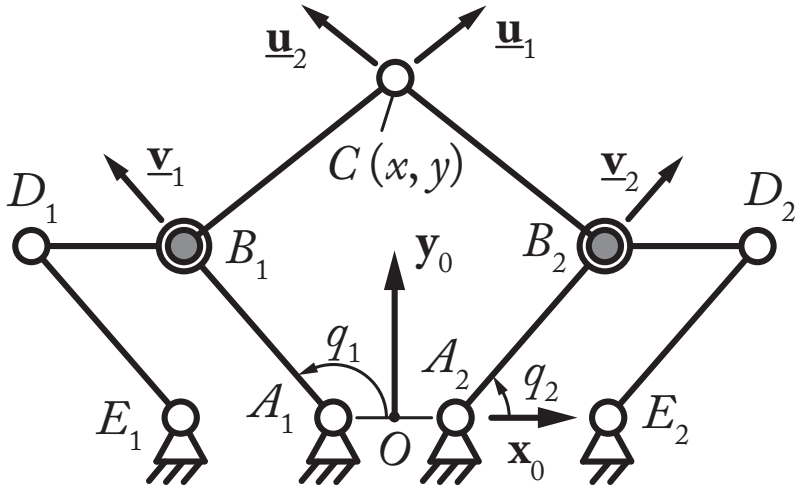



Figure 4.3: The hidden robot involved into the leg-direction-based visual servoing approach of the five-bar mechansim (the gray pairs denote the actuated joints)

which is different from the Type 2 singularities of the five-bar mechanism that appear when the points $B_1$, $B_2$ and $C$ are aligned (Fig. 4.5). As demonstrated in [11], these singularities affect the performance of the controller in terms of accuracy and need to be well handled.

An example of singularity loci in the workspace of a given five-bar mechanism is provided in Fig. 4.6.

### 4.2.2 Singularities of the line-based visual servoing controller

It is known that the singularity conditions appear when the inverse or forward geometric model degenerates. The geometric models involved in this new controller are based on the fact that we can rebuild the end-effector pose by knowing the intersection point between the two lines $\mathcal{L}_1$ and $\mathcal{L}_2$ depicted in Fig 4.1. Therefore, the singularities appear when these two lines are parallel (intersection point at infinity) or coincide (infinity of possible intersection points). Such singularity conditions are equivalent to the Type 2 singularity conditions of the five-bar mechanism (Fig. 4.5(a)).

### 4.2.3 Discussion on the control schemes

At this step, it appears that the new controller has several advantages with respect to the approach proposed in [7] that should be clearly pointed out:

1. contrary to the past approach, the new one does not need the use of the geometric parameters of the robot (except the radius of the observed cylinder) for estimating the platform pose. This is a great advantage because we only need to accurately calibrate the observed cylinders, not the entire robot, for obtaining the best robot accuracy,

2. the singularities of the new controller coincide with those of the real mechanism, which is a great advantage with respect to the past approach, for which the singularities are different and thus lead to the decrease of the reachable workspace.

In the next Section, the two control schemes are compared in terms of robustness to measurement noise in order to clearly demonstrate which type of controller is the best.

## 4.3 Comparative analysis of the controller performance

In order to do a comparative analysis of the two control approaches, it has been created an Adams model of a five-bar mechanism with the following set of parameters: $l_{1i} = 0.3$ m as length of the legs attached to the ground, $l_{2i} = 0.35$ m as length of the legs attached to the end-effector, $l_{OA_i} = 0.275$ m as distance between $O$ and $A_i$ (Fig. 4.1). The workspace is plotted in Fig. 4.6. Both the leg-direction-based controller (case 1) and the line-based controller (case 2) have been applied to such a model.

### 4.3.1 Robustness to measure noise near the leg-direction-based controller singularity

We added noise on the measurements in order to compare the performance of both types of controller. The noise model is described thereafter.
The extraction of the Plücker coordinates of the leg line is based on the equations of the leg edges. In the simulation, they are projected to the image plane and converted from meter to pixel. Then, the edge line intersections with image boundary are computed: the coordinates of the intersection points have to be rounded due to the pixel accuracy. A new equation of the edge line is then recomputed taking into account the error introduced in the intersection points between the edge line and the image boundary.
In this subsection the results of the leg-direction-based and the line-based visual servoing approaches subjected to measurement noise are shown. The measurements error chosen is given by a pixel accuracy equal to 1. Then, it has been chosen the initial end-effector pose as $(x_0, y_0) = (0, 0.196)m$ and a set of desired positions $\mathbf{C}_d$ of the end-effector near the singularity of the hidden robot of the leg-direction-based controller:
$\mathbf{C}_{d1} = (-0.172, 0.030)m$
$\mathbf{C}_{d2} = (-0.050, 0.080)m$
$\mathbf{C}_{d3} = (0.036, 0.082)m$
$\mathbf{C}_{d4} = (0.092, 0.070)m$
$\mathbf{C}_{d5} = (0.193, 0.013)m$

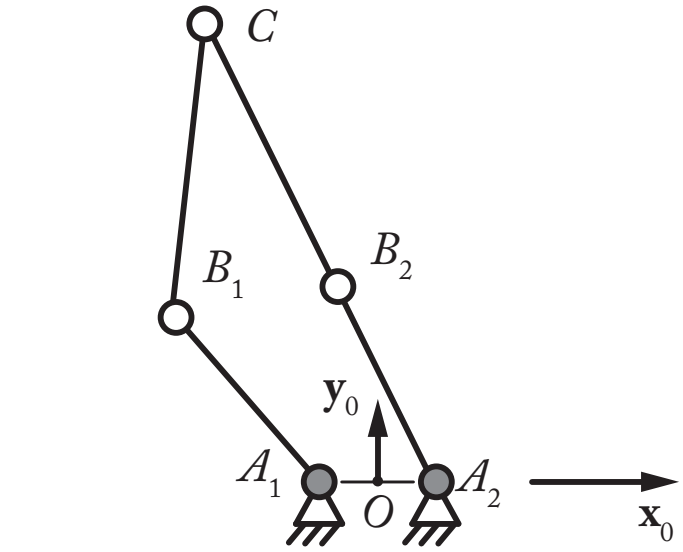
The initial position, the desired positions and the final positions got with the controller of case 2 are shown inside the plot of the workspace in Fig. 4.6 (from left to right, in black: $\mathbf{C}_{d1}...\mathbf{C}_{d5}$). The results of all the simulations are then shown in the Table (4.3.1): for each desired position, the final position and the error got with both the controller are shown. In the Table (4.3.1): $\mathbf{C}_d$ is the set of desired positions, $\mathbf{C}_{f1}$ and $\mathbf{C}_{f2}$ are the set of the final positions in the controllers of case 1 and case 2 respectively, $e_1(t_f)$ and $e_2(t_f)$ are the errors of the controllers of case 1 and case 2 respectively. The error is computed as the norm of the difference between the final position (the final time chosen is $t_f = 3s$) and the desired position. The graphs on Figs. 5.26, 5.33, 5.29, 5.30, 4.11 show the convergence of the end-effector pose of both controller in the cases $\mathbf{C}_{d1}, \mathbf{C}_{d2}, \mathbf{C}_{d3}, \mathbf{C}_{d4}, \mathbf{C}_{d5}$.

Upon the results, it is readily found that the singularity of the hidden robot of the controller of case 2 is much more robust to measurement noise, which allows to access, with this new controller, the same workspace zones as the real robot (contrarily to the controller of case 1).

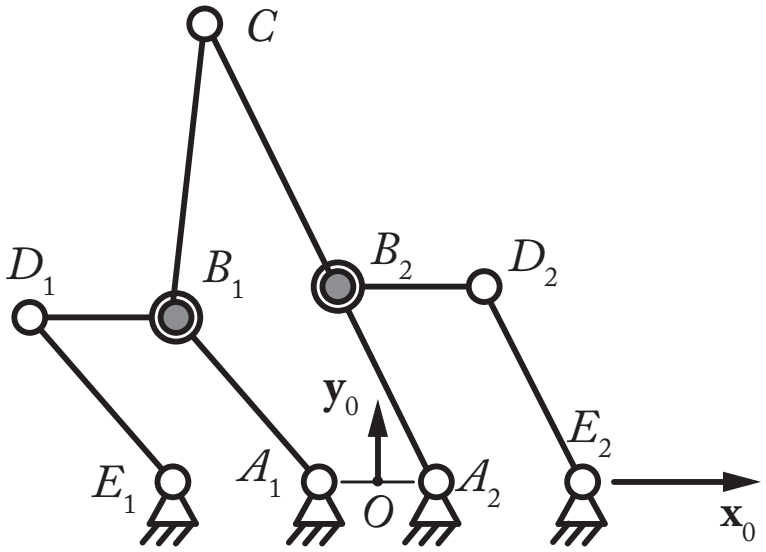
| $\mathbf{C}_d$ | $\mathbf{C}_{f1}$ | $\mathbf{C}_{f2}$ | $e_1(t_f)$ | $e_2(t_f)$ |
|---|---|---|---|---|
| (-0.172, 0.030) | (-0.157, 0.040) | (-0.173, 0.031) | 0.0184 | 0.0013 |
| (-0.05, 0.080) | (-0.027, 0.083) | (-0.050, 0.080) | 0.0232 | 0.0004 |
| (0.036, 0.082) | (0.061, 0.078) | (0.036, 0.083) | 0.0255 | 0.0007 |
| (0.092, 0.070) | (0.107, 0.065) | (0.092, 0.070) | 0.0158 | 0.0003 |
| (0.193, 0.013) | (0.227, -0.020) | (0.193, 0.013) | 0.0471 | 0.0001 |

### 4.3.2 Crossing the hidden robot singularity

In this subsection, the end-effector desired position is chosen in such a way that the end-effector should cross the hidden robot singularity (singularity of the controller of case 1) shown in the Fig. 4.6. It is shown that in the case of leg-direction-based approach the legs direction converges to the desired one, but the end-effector position does not. While in the case of the line-based approach also the end-effector pose converges to the desired one. This is due to the fact that, in the controller of case 1, the five-bar mechanism converges to another assembly mode of its hidden robot.
The end-effector initial position is the same of the subsection (4.3.1), while the chosen desired position is $\mathbf{C}_d = (0.104, 0.036)m$. The results of the simulations are shown in Fig. 4.12.

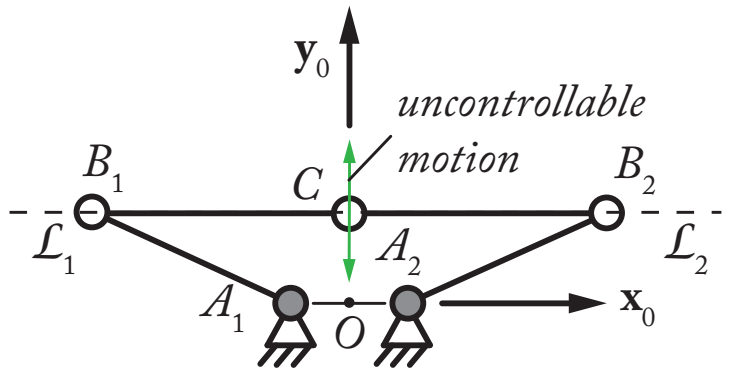



(a)

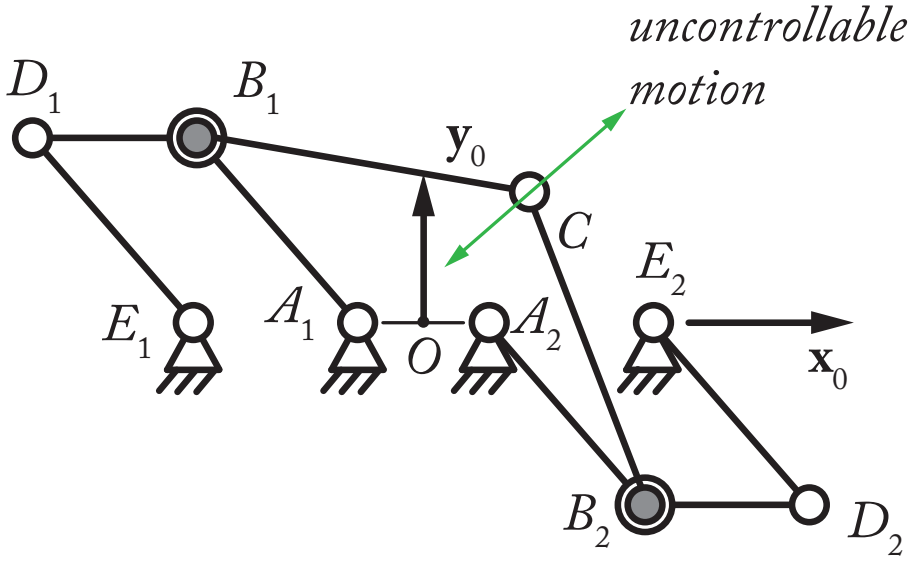



(b)

Figure 4.4: Examples of Type 1 singularity for the five-bar mechanism (a) and its corresponding hidden robot (b)




(a)




(b)

Figure 4.5: Examples of Type 2 singularity for the five-bar mechanism (a) and its corresponding hidden robot (b)

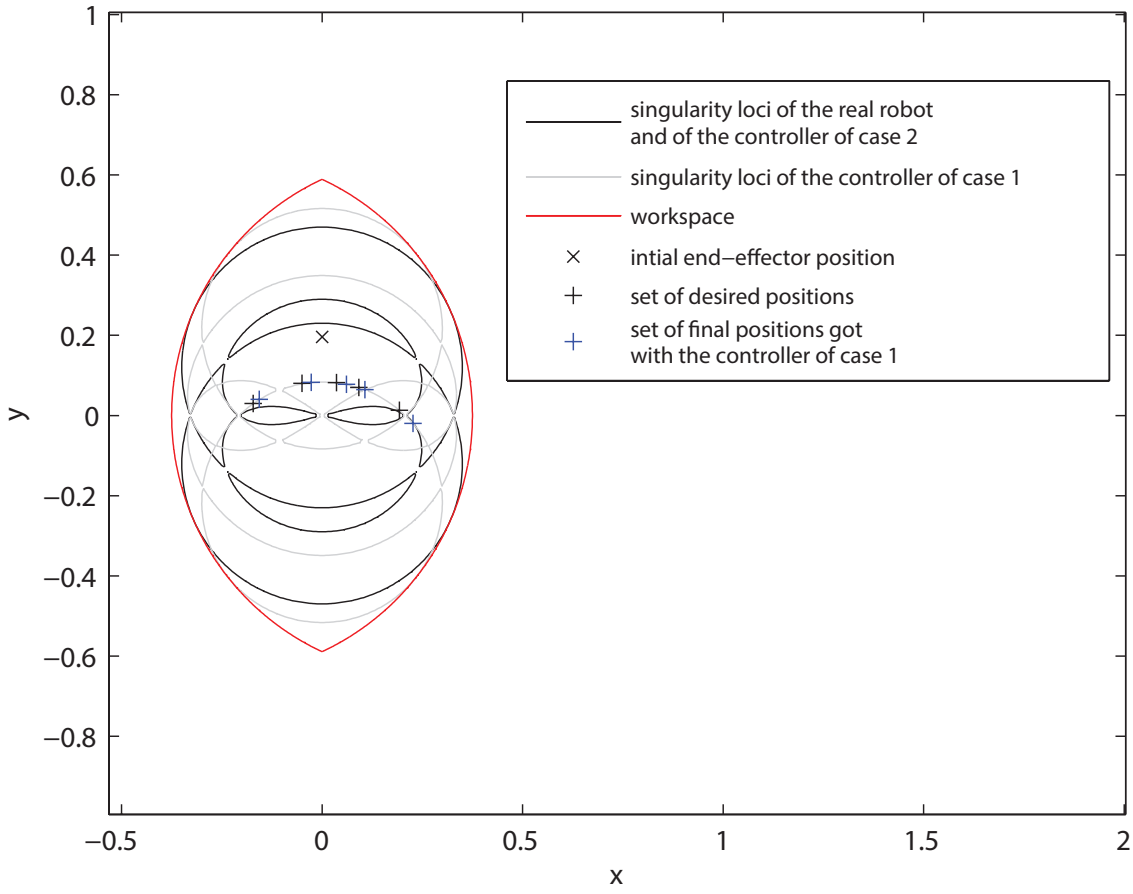
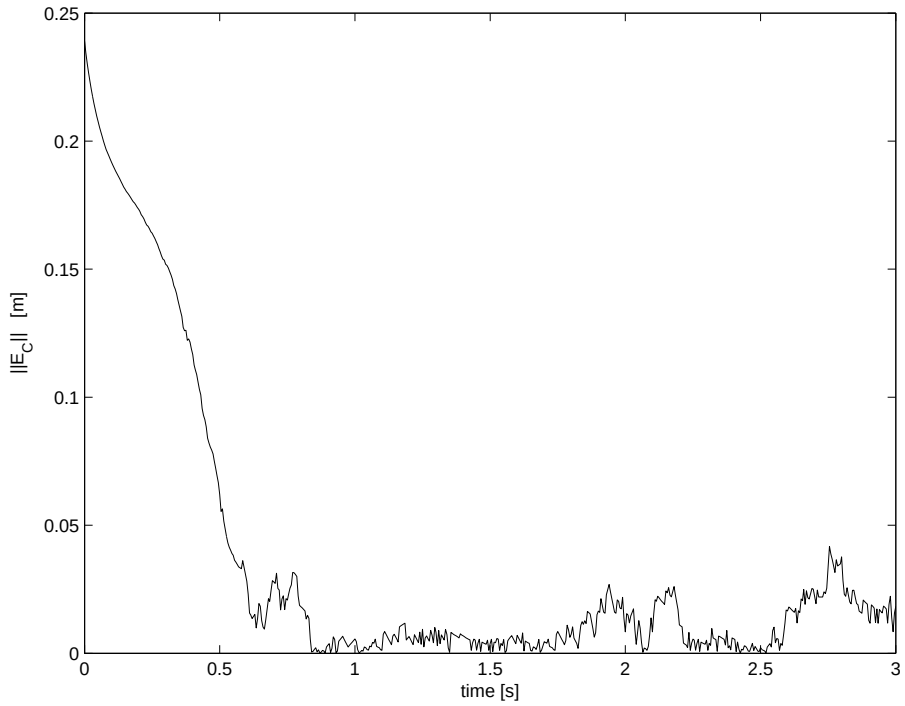
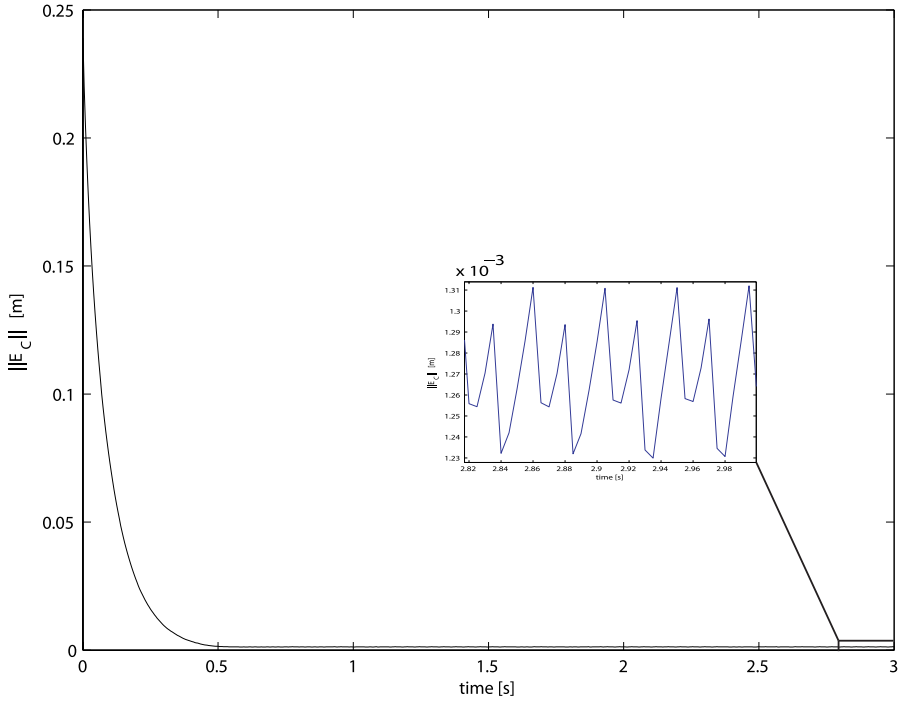



Figure 4.6: Singularity loci of the five-bar mechanism and its corresponding hidden robot for the following set of parameters: $l_{1i} = 0.3$ m, $l_{2i} = 0.35$ m, $l_{OA_i} = 0.275$ m.

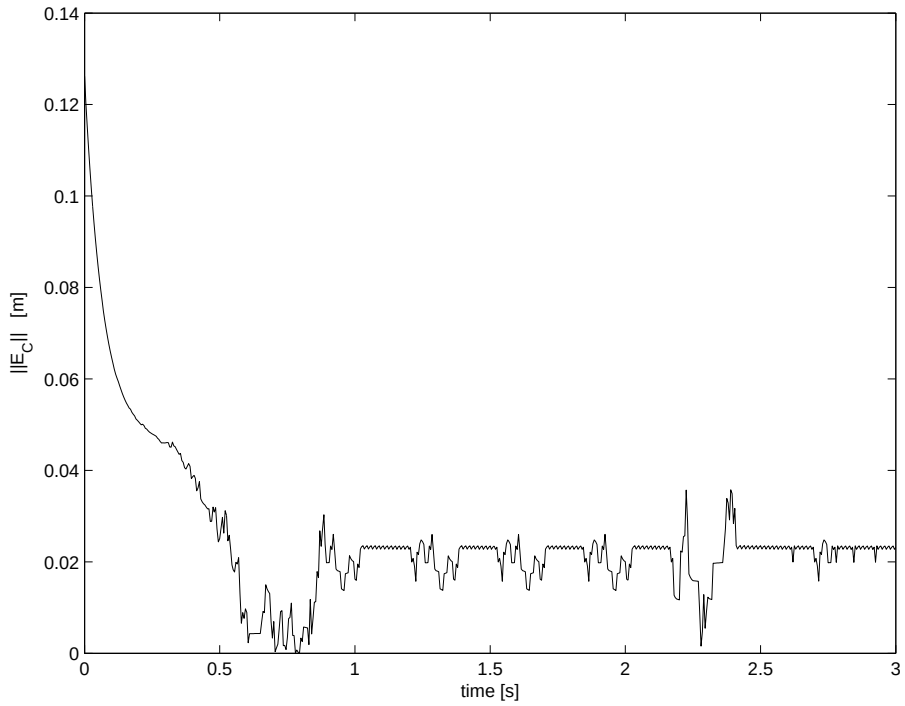



(a) `End-effectorposeerrorintheleg-direction-basedcontroller.`

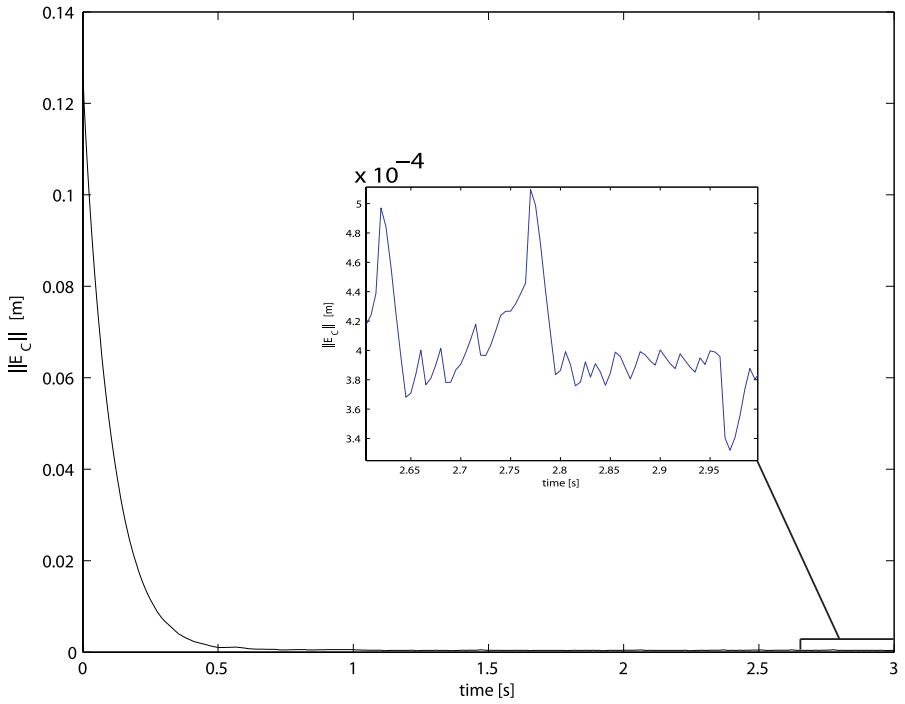



(b) `End-effectorposeerrorintheline-basedcontroller.`

Figure 4.7: Error in the case of measurement noise with desired position near the singularity of the leg-direction-based controller: $\mathbf{C}_{d1} = (-0.172, 0.030)m$

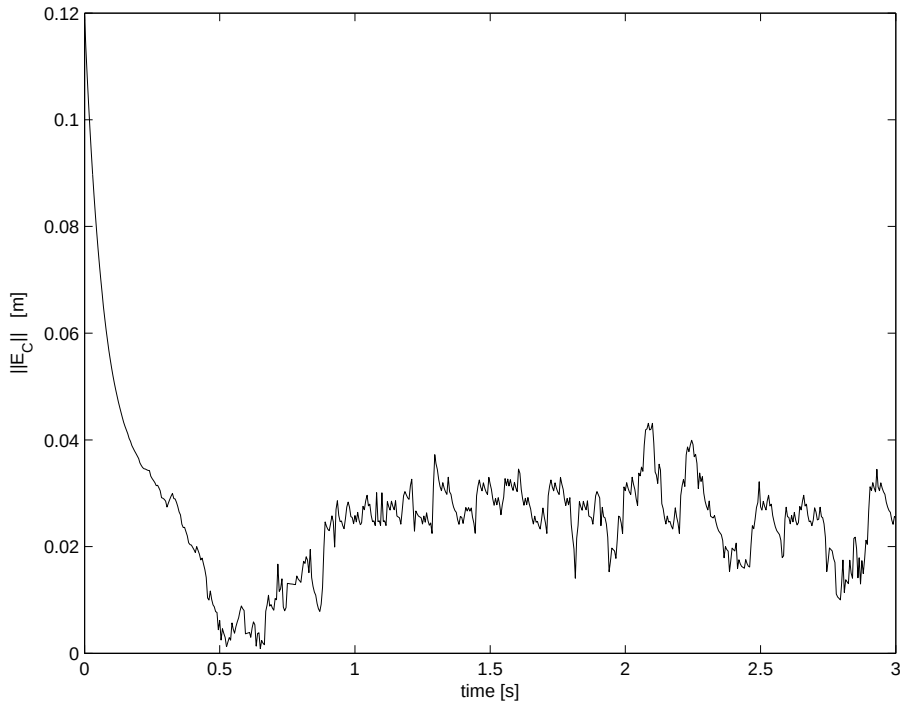



(a) `End-effectorposeerrorintheleg-direction-basedcontroller.`

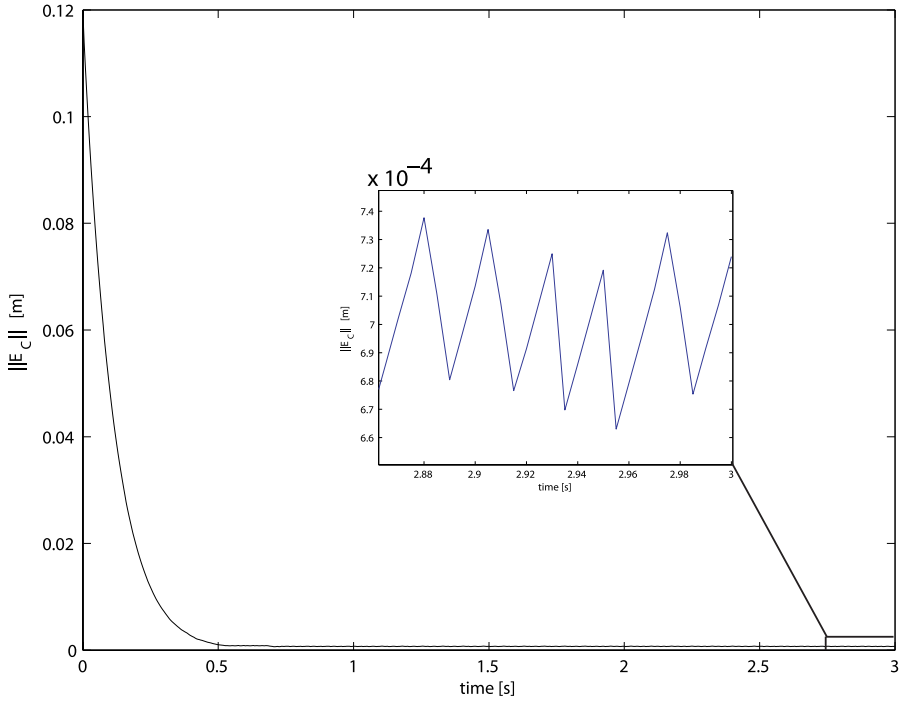



(b) `End-effectorposeerrorintheline-basedcontroller.`

Figure 4.8: Error in the case of measurement noise with desired position near the singularity of the leg-direction-based controller: $\mathbf{C}_{d2} = (-0.050, 0.080)m$

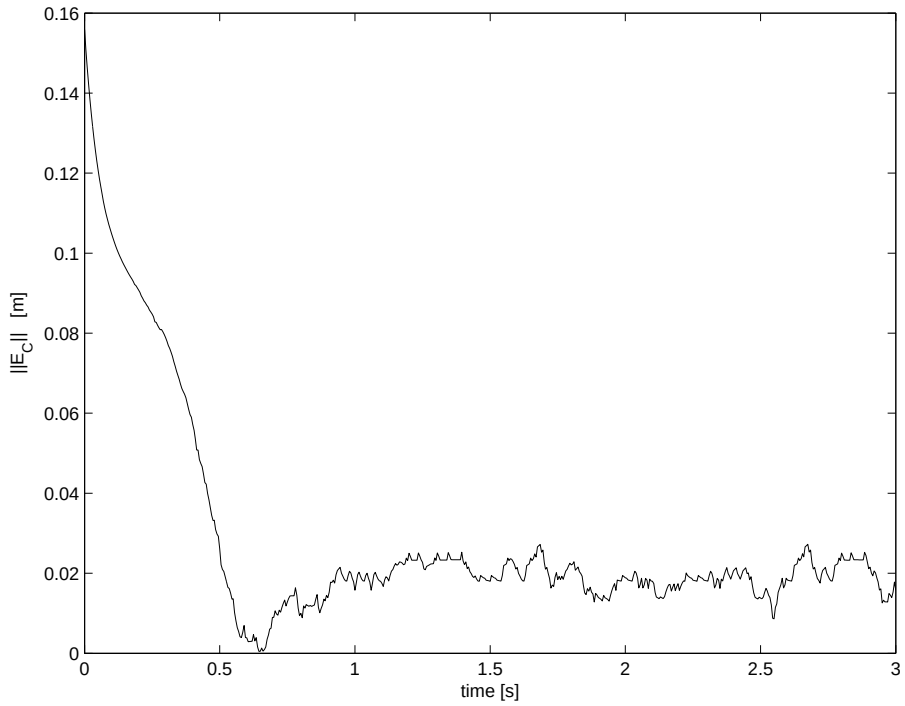



(a) `End-effectorposeerrorintheleg-direction-basedcontroller.`

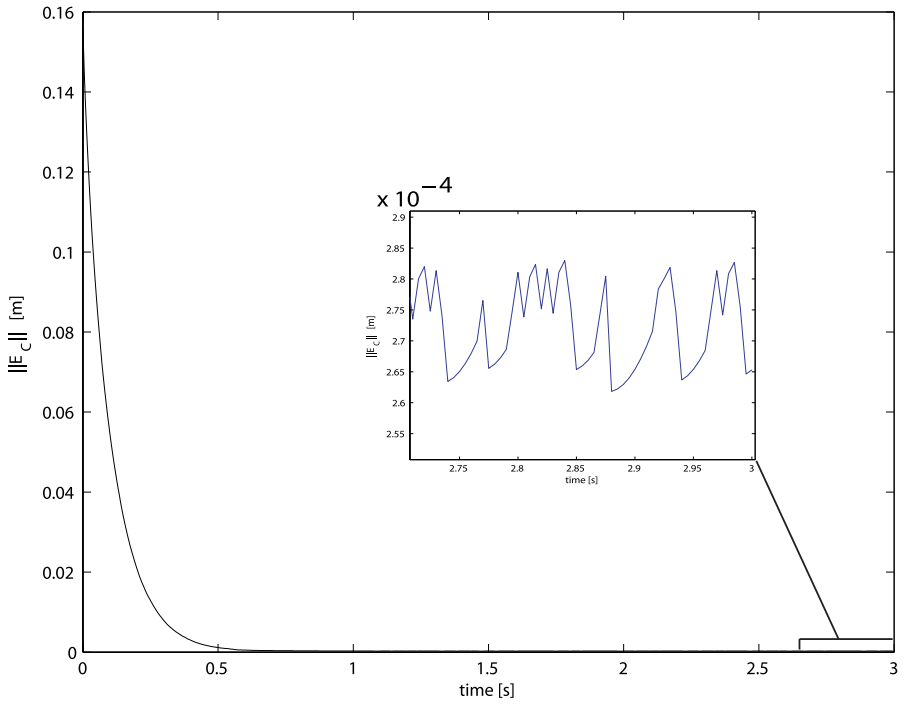



(b) `End-effectorposeerrorintheline-basedcontroller.`

Figure 4.9: Error in the case of measurement noise with desired position near the singularity of the leg-direction-based controller: $\mathbf{C}_{d3} = (0.036, 0.082)m$

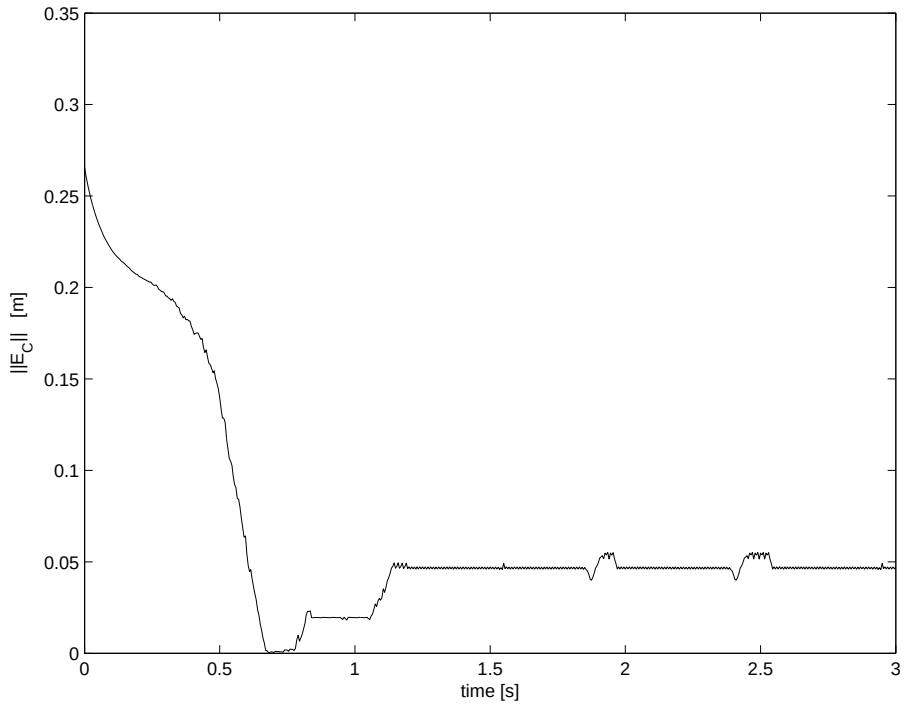



(a) `End-effectorposeerrorintheleg-direction-basedcontroller.`

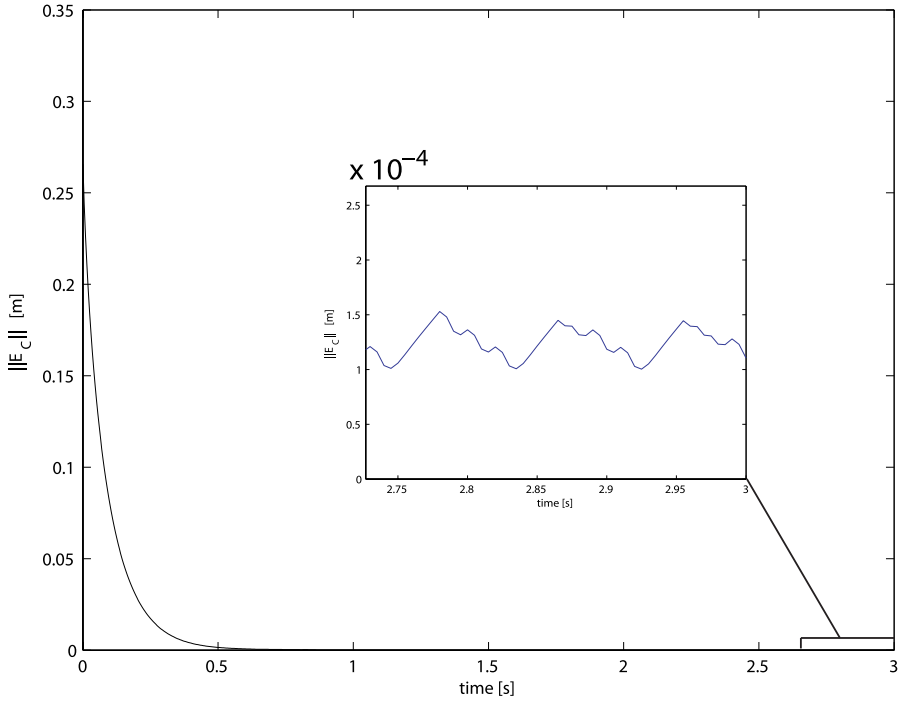



(b) `End-effectorposeerrorintheline-basedcontroller.`

Figure 4.10: Error in the case of measurement noise with desired position near the singularity of the leg-direction-based controller: $\mathbf{C}_{d4} = (0.092, 0.070)m$

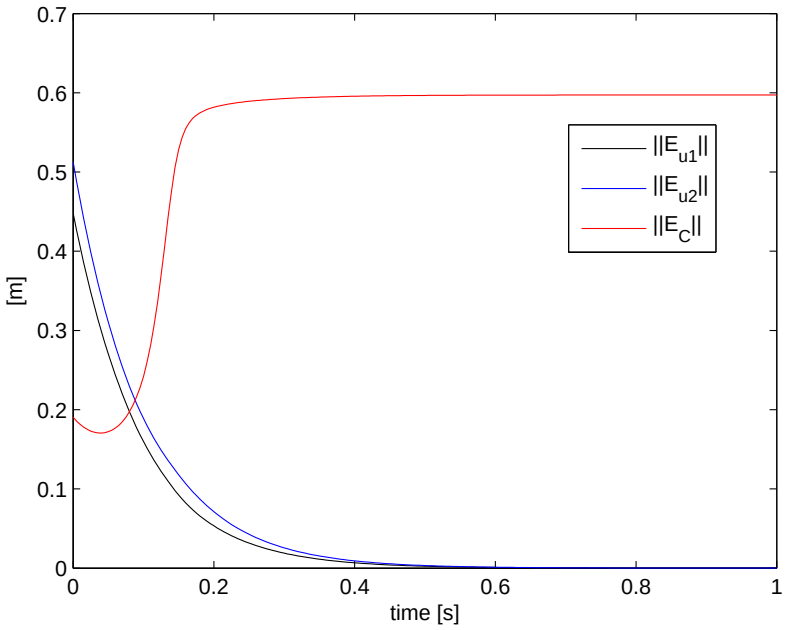



(a) `End-effectorposeerrorintheleg-direction-basedcontroller.`

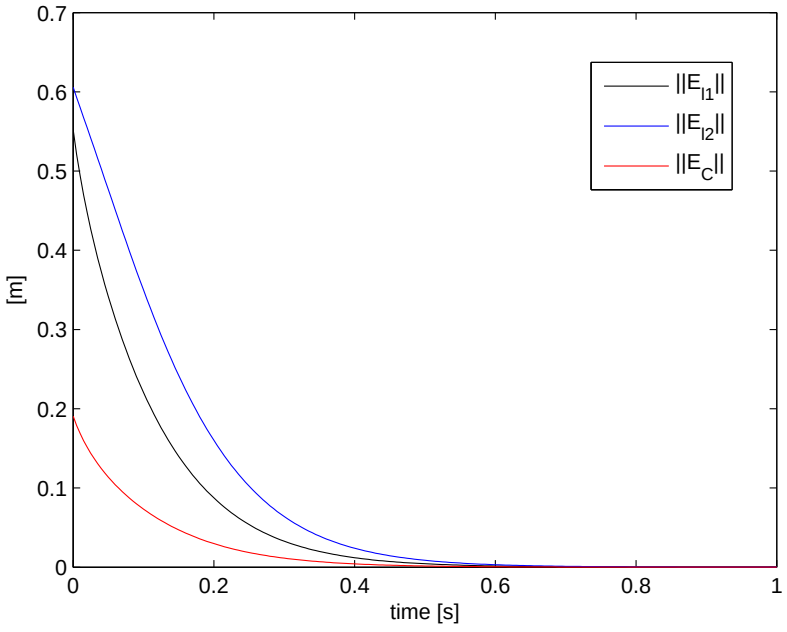



(b) `End-effectorposeerrorintheline-basedcontroller.`

Figure 4.11: Error in the case of measurement noise with desired position near the singularity of the leg-direction-based controller: $\mathbf{C}_{d5} = (0.193, 0.013)m$




(a) `Errorsoftheleg-direction-basedcontroller.`




(b) `Errorsoftheline-basedcontroller.`

Figure 4.12: Crossing the hidden robot singularity: $\mathbf{C}_d = (0.104, 0.036)m$

# Chapter 5

# Line-based visual servoing of the MEPaM

## 5.1 Design of MEPaM

There are a lot of parallel manipulators which have been developed during the past three decades. Among them, there are parallel manipulators with 3-*dof*, 4-*dof*, 5-*dof* and 6-*dof*. 6-*dof* parallel manipulators with six legs, such as the Gough–Stewart (GS) platform, have high stiffness and accuracy but a small workspace and a frequent limb interference.
In order to overcome these limitations, 6-*dof* manipulators with three legs were introduced. To achieve six-*dof* with only three legs requires actuators to be mounted on the moving limbs, thus increasing the mass and inertia of the moving parts. Some 6-*dof* manipulators with three legs have been proposed, [30][31][32][33][34][35][36] but most of them have actuators not allocated on the ground.

MEPaM has an innovative design (Fig. 5.1). The actuators are mounted on the base instead of the moving limbs, reducing their mass and inertia. It can preserve six-*dof* thanks to planetary-belt systems, which transmit power to the moving legs. There are three planetary-belt systems, and each one provides two-*dof* in a plane and is driven by two motors: a lower motor drives the carrier A via a short stiff belt, and a upper motor drives the sun pulley via a long stiff belt. The planetary-belt system ends with the lever arm B, whose end is attached to a cylindrical joint perpendicular to the driving plane of the system. Then, one universal joint links every cylindrical joint to a vertex of the triangular end-effector.

## 5.2 Model in Adams

As for the five-bar mechanism, the simulator of the MEPaM has been developed in Adams environment. The plant inputs are the joint velocities, while the outputs are the coordinates of two points for each of the three legs and the angular coordinates of the three actuators $q_1$, $q_2$ and $q_3$.

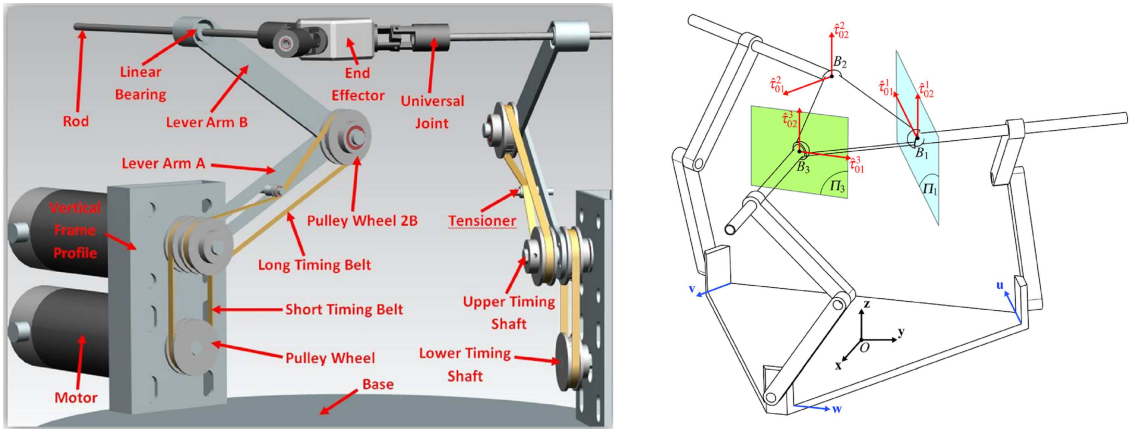



Figure 5.1: Left: Virtual model of MEPaM (one leg is hidden for clarity); Right: The actuation forces applied on the moving platform of MEPaM

## 5.3 Visual servoing of MEPaM

### 5.3.1 Forward kinematics based on the actuators angles

The following equations and the Fig. 5.2 are useful to introduce the kinematics of MEPaM, already analyzed in [37].

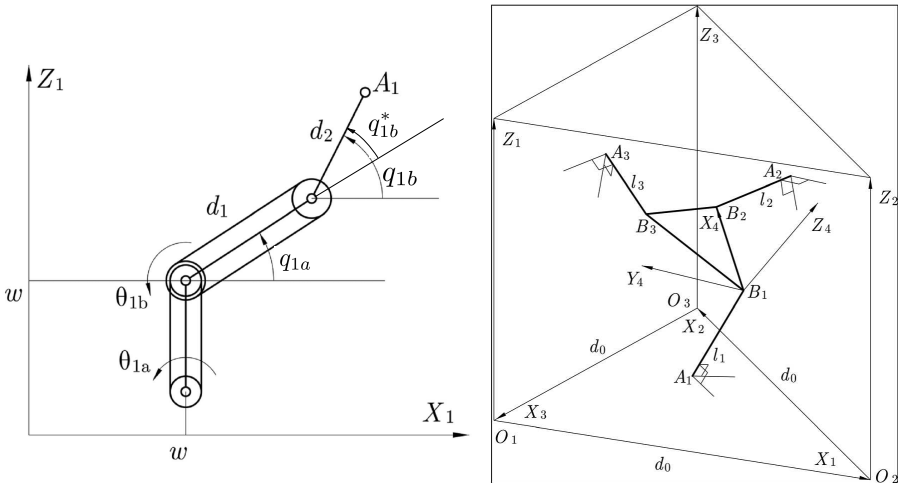



Figure 5.2: Left: The belt-pulley transmission of MEPaM; Right: The end-effector of MEPaM connected to $A_1$, $A_2$ and $A_3$

MEPaM has six motors, which correspond to six angles: $\theta_{1a}, \theta_{1b}, \theta_{2a}, \theta_{2b}, \theta_{3a}$ and $\theta_{3b}$ of the Fig. 5.2. Then, there are the angles of the carrier and the planet, which are $q_{1a}, q_{1b}, q_{2a}, q_{2b}, q_{3a}$ and $q_{3b}$. As we can see on the left of the Fig. 5.2, the Cartesian coordinates of $A_1$ are

$$^1A_{1x} = w + d_1 cosq_{1a} + d_2 cosq_{1b} \tag{5.1}$$

$$^1A_{1z} = w + d_1 sinq_{1a} + d_2 sinq_{1b} \tag{5.2}$$

where $(w, w)$ are the Cartesian coordinates of the sun center. Due to the planetary transmission, we have

$$\frac{\theta_{1b} - q_{1a}}{q_{1b} - q_{1a}} = 1 \tag{5.3}$$

Chosen a proper reference for $\theta_{1a}$, we have $q_{1a} = \theta_{1a}$. So, the equation (5.3) becomes $q_{1b} = \theta_{1b}$. Then, the carrier and the planet motions can be controlled independently by motor 1A and 1B. Eqs. (5.1) and (5.2) become

$$^{i}A_{ix} = w + d_1 cos\theta_{ia} + d_2 cos\theta_{ib} \tag{5.4}$$

$$^{i}A_{iz} = w + d_1 sin\theta_{ia} + d_2 sin\theta_{ib} \tag{5.5}$$

for $i = 1, 2, 3$.

The frame assignment for the driving planes as well as the end-effector is shown in Fig. 5.2 (right). Three fixed frames, $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$. are attached to the base in an equilateral triangle formation, while a moving frame $\mathcal{F}_4$ is attached to the triangular end-effector of side length $d_3$. $\mathbf{a}_i$ and $\mathbf{b}_i$ are the Cartesian coordinate vectors of points $A_i$ and $B_i$, respectively. An upper-left index is used to indicate in which frame the vector is expressed. Since each cylindrical joint is perpendicular to the corresponding plane, we have

$$^{1}\mathbf{b}_1 = \begin{bmatrix} ^{1}A_{1x} \\ l_1 \\ ^{1}A_{1z} \end{bmatrix}, ^{2}\mathbf{b}_2 = \begin{bmatrix} ^{2}A_{2x} \\ l_2 \\ ^{2}A_{2z} \end{bmatrix}, ^{3}\mathbf{b}_3 = \begin{bmatrix} ^{3}A_{3x} \\ l_3 \\ ^{3}A_{3z} \end{bmatrix} \tag{5.6}$$

From Fig.5.2, the transformation matrices between frames $\mathcal{F}_1$, $\mathcal{F}_2$ and $\mathcal{F}_3$ are given by

$$^{1}_{2}\mathbf{T} = ^{2}_{3}\mathbf{T} = ^{3}_{1}\mathbf{T} = \begin{bmatrix} cos(2\pi/3) & -sin(2\pi/3) & 0 & d_0 \\ sin(2\pi/3) & cos(2\pi/3) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.7}$$

The position of all the vertices of the platform can be transformed into $\mathcal{F}_1$,

$$^{1}b_i = ^{1}_{i}\mathbf{T}\, ^{i}b_i \tag{5.8}$$

for $i = 2, 3$. The geometric constraints on the device are given by

$$\left\| ^{1}b_i - ^{1}b_j \right\| = d_3^2 \tag{5.9}$$

for $i = 1, 2, 3$ and $j = i + 1 \pmod 3$.

The constraint equation, Eq. 5.9, contain only three variables $l_i, i = 1, 2, 3$, and can be written in the form

$$D_2 l_2^2 + D_1 l_2 + D_0 = 0 \tag{5.10}$$

$$E_2 l_3^2 + E_1 l_3 + E_0 = 0 \tag{5.11}$$

$$F_2 l_3^2 + F_1 l_3 + F_0 = 0 \tag{5.12}$$

where $D_j, E_j, F_j (j = 0, 1, 2)$ arefunctions of $l_1, l_2$ and $l_1$, respectively, as well as $d_3$, $A_{ix}$ and $A_{iz}$ $(i = 1, 2, 3)$. By means of dialytic elimination, the system of equations 5.10, 5.11, 5.12 can be reduced to a univariate polynomial of order four in the variable $l_1$

$$G_4 l_1^4 + G_3 l_1^4 + G_2 l_1^2 + G_1 l_1^2 + G_0 = 0 \tag{5.13}$$

Once Eq. 5.13 has been solved for $l_1$, substitution of the result into Eqs. 5.10, 5.11, 5.12 will allow for determination of $l_2$ and $l_3$. The complexity of Eqs. 5.9 is relatively low as compared to the direct kinematics problem in most six-*dof* parallel manipulators. Solutions of $l_i$ for $i = 1,2,3$ can be used to evaluate the position and orientation of the end-effector. The position of the end-effector is simply the origin of $\mathcal{F}_4$, given by ${}^1\mathbf{b}_1 = [A_{1x}, l_1, A_{1z}]^T$ The orientation of the end-effector is given by ${}^1_4\mathbf{Q} = [\mathbf{i}, \mathbf{j}, \mathbf{k}]$ where

$$\mathbf{i} = \frac{{}^1\mathbf{b}_2 - {}^1\mathbf{b}_1}{\|{}^1\mathbf{b}_2 - {}^1\mathbf{b}_1\|} \tag{5.14}$$

$$\mathbf{j} = \frac{({}^1\mathbf{b}_3 - {}^1\mathbf{b}_1) - \mathbf{i}\mathbf{i}^T({}^1\mathbf{b}_3 - {}^1\mathbf{b}_1)}{\left\|({}^1\mathbf{b}_3 - {}^1\mathbf{b}_1) - \mathbf{i}\mathbf{i}^T({}^1\mathbf{b}_3 - {}^1\mathbf{b}_1)\right\|} \tag{5.15}$$

$$\mathbf{k} = \mathbf{i} \times \mathbf{j} \tag{5.16}$$

The direct kinematics of MEPaM is solved.

### 5.3.2 Forward kinematics based on Plücker coordinates

The objective of the thesis is to use the information of the Plücker coordinates instead of the joint angles, therefore the forward kinematic problem has to be solved using them. The method will be the same of subsection 5.3.1, but ${}^iA_i$ is computed using the Plücker coordinates of the legs.

For finding ${}^iA_i$ as a function of $(\underline{\mathbf{u}}_i, \mathbf{h}_i)$, the following formula has to be applied,

$${}^iA_i = {}^i_0\,T \cdot {}^0\,A_i \tag{5.17}$$

with ${}^0A_i$ which is given by the intersection between the $i$th leg and the $i$th plane (Eq. 5.18).

The general formula which gives the point of intersection between a line and a plane, both expressed in Plücker coordinates, is

Plane:
$ax + by + cz + dw = 0$ with $d = 1$
$[N : n] = [(a : b : c) : d]$

Line:
$[\underline{\mathbf{u}}, \mathbf{h}]$

Point of intersection:

$$A_i = (-\mathbf{h} \times N - n \cdot \underline{\mathbf{u}} : \underline{\mathbf{u}} \cdot N) \tag{5.18}$$

### 5.3.3 Line-based visual servoing of the MEPaM

The part concerning the simulation of the camera, the extraction of the Plücker coordinates, the visual primitive, the error and the control law, are the same of the five-bar mechanism case (subsection 4.1.3) with the only difference that the legs to be considered are three instead of two. Furthermore, in this case the point to be controlled is not simply the interaction point of the legs, then the interaction matrix is computed differently than in the case of the five-bar mechanism.

### 5.3.4 Interaction matrix

The matrix $\mathbf{M}_l^T$ can be obtained deriving the following equations

$$ {}^iB_{ix}(x,y,z,\phi_1,\phi_2,\phi_3) - {}^iA_{ix}(\underline{\mathbf{u}_1},\underline{\mathbf{u}_2},\underline{\mathbf{u}_3},\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3) = 0 \tag{5.19} $$

$$ {}^iB_{iz}(x,y,z,\phi_1,\phi_2,\phi_3) - {}^iA_{iz}(\underline{\mathbf{u}_1},\underline{\mathbf{u}_2},\underline{\mathbf{u}_3},\mathbf{h}_1,\mathbf{h}_2,\mathbf{h}_3) = 0 \tag{5.20} $$

with $(x,y,z,\phi_1,\phi_2,\phi_3) = \mathbf{X}$ are the position coordinates and the Roll-Pitch-Yaw angles of the platform, with $i = 1,2,3$ and the other terms as below.

${}^iA_i$ and ${}^iB_i$ are computed with respect to the $i$th frame, shown in the Fig. 5.3.

For finding ${}^iB_i$, the formula is

$$ {}^iB_i = {}^i_E T \cdot {}^E B_i = {}^i_0 T \cdot {}^0_E T \cdot {}^E B_i \tag{5.21} $$

where ${}^0_E T =$ is given by a rotation $R = R(z,\phi_1)R(y,\phi_2)R(x,\phi_3)$ (Roll-Pitch-Yaw angles) and a translation of $(x,y,z)$.

The translational matrices are

$$ {}^0_E T = \begin{bmatrix} \mathrm{C}\phi_1\mathrm{C}\phi_2 & \mathrm{C}\phi_1\mathrm{S}\phi_2\mathrm{S}\phi_3 - \mathrm{S}\phi_1\mathrm{C}\phi_3 & \mathrm{C}\phi_1\mathrm{S}\phi_2\mathrm{C}\phi_3 + \mathrm{S}\phi_1\mathrm{S}\phi_3 & x \\ \mathrm{S}\phi_1\mathrm{C}\phi_2 & \mathrm{S}\phi_1\mathrm{S}\phi_2\mathrm{S}\phi_3 + \mathrm{C}\phi_1\mathrm{C}\phi_3 & \mathrm{S}\phi_1\mathrm{S}\phi_2\mathrm{C}\phi_3 - \mathrm{C}\phi_1\mathrm{S}\phi_3 & y \\ -\mathrm{S}\phi_2 & \mathrm{C}\phi_2\mathrm{S}\phi_3 & \mathrm{C}\phi_2\mathrm{C}\phi_3 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.22} $$

$$ {}^1_0 T = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & r1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.23} $$

$$ {}^2_0 T = \begin{bmatrix} \mathrm{C}(\frac{7}{6}\pi) & \mathrm{S}(\frac{7}{6}\pi) & 0 & \frac{r2}{2}\mathrm{C}(\frac{7}{6}\pi) - \sqrt{3}\frac{r2}{2}\mathrm{S}(\frac{7}{6}\pi) \\ -\mathrm{S}(\frac{7}{6}\pi) & \mathrm{C}(\frac{7}{6}\pi) & 0 & -\frac{r2}{2}\mathrm{S}(\frac{7}{6}\pi) - \sqrt{3}\frac{r2}{2}\mathrm{C}(\frac{7}{6}\pi) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.24} $$

$$ {}^3_0 T = \begin{bmatrix} \mathrm{C}(\frac{11}{6}\pi) & \mathrm{S}(\frac{11}{6}\pi) & 0 & \frac{r3}{2}\mathrm{C}(\frac{11}{6}\pi) + \sqrt{3}\frac{r3}{2}\mathrm{S}(\frac{11}{6}\pi) \\ -\mathrm{S}(\frac{11}{6}\pi) & \mathrm{C}(\frac{11}{6}\pi) & 0 & -\frac{r3}{2}\mathrm{S}(\frac{11}{6}\pi) + \sqrt{3}\frac{r3}{2}\mathrm{C}(\frac{11}{6}\pi) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.25} $$

The Cartesian coordinates of the vertices of the platform with respect to the platform frame (Fig. 5.4) are

$$ {}^E B_1 = \begin{bmatrix} 0.043301300000000 \\ 0 \\ 0 \end{bmatrix}, \; {}^E B_2 = \begin{bmatrix} -0.021650650000000 \\ 0.037500025816891 \\ 0 \end{bmatrix}, \; {}^E B_3 = \begin{bmatrix} -0.021650650000000 \\ -0.037500025816891 \\ 0 \end{bmatrix} $$

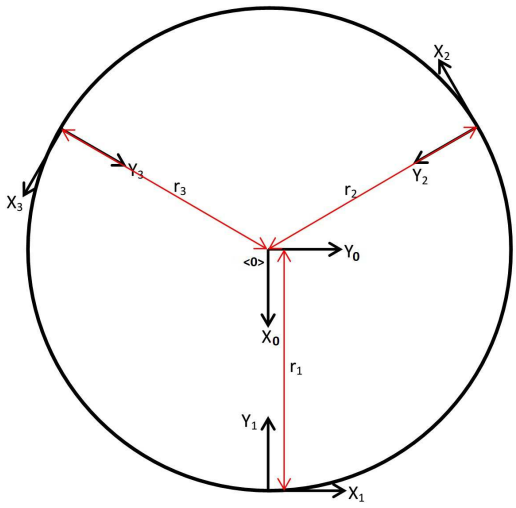



Figure 5.3: The three frames of the three legs. ${}^{i}A_i$ and ${}^{i}B_i$ are calculated with respect to the $i$th frame.

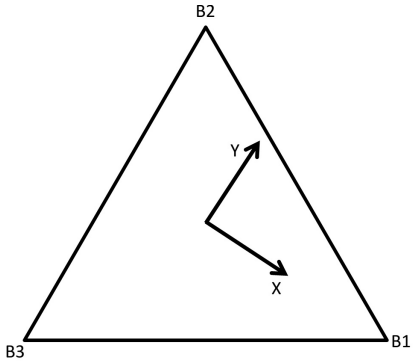



Figure 5.4: Platform frame.

For finding ${}^{i}A_i$ as a function of $(\underline{\mathbf{u}}_i, \mathbf{h}_i)$, the Eq. 5.17 has to be used.

Deriving Eq. 5.17, I obtained

$$A \cdot \tau_p + P \cdot \dot{\mathbf{l}} = 0 \tag{5.26}$$

and therefore the interaction matrix

$$M^T = -P^+ \cdot A \tag{5.27}$$

where $P$ is

$$P = \begin{bmatrix} -h_{2z} & 0 & 0 & h_{1z} & 0 & 0 & 0 & 0 & u_{2x} & 0 \\ 0 & -u_{1x} & & & & & & & & \\ 0 & -h_{2z} & 0 & 0 & h_{1z} & 0 & 0 & 0 & u_{2y} & 0 \\ 0 & -u_{1y} & & & & & & & & \\ 0 & 0 & 0 & -h_{1x} & -h_{1y} & 0 & -u_{2x} & -u_{2y} & 0 & 0 \\ 0 & 0 & & & & & & & & \\ -u_{2y} & u_{2x} & 0 & u_{1y} & -u_{1x} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & & & & & & & & \end{bmatrix} \tag{5.28}$$

### 5.3.5 Inverse Jacobian

For finding the inverse Jacobian, the following equations have to be derived.

$$ {}^{i}B_{ix}(x,y,z,\phi_1,\phi_2,\phi_3) - {}^{i}A_{ix}(q_{1a},q^*_{1b},q_{2a},q^*_{2b},q_{3a},q^*_{3b}) = 0 \tag{5.29} $$

$$ {}^{i}B_{iz}(x,y,z,\phi_1,\phi_2,\phi_3) - {}^{i}A_{ix}(q_{1a},q^*_{1b},q_{2a},q^*_{2b},q_{3a},q^*_{3b}) = 0 \tag{5.30} $$

Please note that ${}^{i}B_{ix}$ and ${}^{i}B_{iz}$ are the same of Eq.5.21, while ${}^{i}A_{i}$ is as follows

$$ {}^{i}A_{ix} = [wx_i + d_{1i}cos(q_{ia}) + d_{2i}cos(q_{ia}+q^*_{ib})] \tag{5.31} $$

$$ {}^{i}A_{iz} = [wz_i + d_{1i}sin(q_{ia}) + d_{2i}sin(q_{ia}+q^*_{ib})] \tag{5.32} $$

Deriving Eq.5.29, it leads to

$$ A\cdot\dot{\mathbf{x}} + B\cdot\dot{\mathbf{q}} = 0 \tag{5.33} $$

$$ Jinv = -B^{-1}\cdot A \tag{5.34} $$

## 5.4 Serial and parallel singularities of the real robot

The serial and parallel singularities can be found from the determinant of the Jacobian matrices of serial and parallel manipulators [38]. The Jacobian matrices are related as follows

$$ \mathbf{J}_S\dot{\Theta} = \mathbf{J}_P\mathbf{t} \tag{5.35} $$

where $\mathbf{J}_S$ is the Jacobian matrix of serial manipulators, $\mathbf{J}_P$ is the Jacobian matrix of parallel manipulators, $\dot{\Theta}$ is the vector of active joints rates and $\mathbf{t}$ is the twist of the moving platform. The serial singularity is found from the Jacobian matrix of serial manipulators which is obtained by differentiating the Eqs. (5.4) and (5.5) with respect to time

$$ \mathbf{J}_s = \begin{bmatrix} \mathbf{J}_1 & \mathbf{0}_{2X2} & \mathbf{0}_{2X2} \\ \mathbf{0}_{2X2} & \mathbf{J}_2 & \mathbf{0}_{2X2} \\ \mathbf{0}_{2X2} & \mathbf{0}_{2X2} & \mathbf{J}_3 \end{bmatrix}, \text{ where } \mathbf{J}_i = \begin{bmatrix} -d_1 sin\theta_{ia} & -d_2 sin\theta_{ib} \\ d_1 cos\theta_{ia} & d_2 cos\theta_{ib} \end{bmatrix} $$

Putting the determinant of $\mathbf{J}_S$ equal to 0 as follows

$$ (d_1 d_2)^3 \prod_{i=1}^{3} sin(\theta_{ia} - \theta_{ib}) = 0 \tag{5.36} $$

we find $\theta_{ia} - \theta_{ib} = 0$ or $\pi$.

Then, there is serial singularity when the arms are fully extended or folded: in such a configuration, the triangular end effector is not able to move in the arms direction, and it loses one-*dof*. Concerning the parallel singularities, in [37] they were determined from geometric conditions derived by Grassmann – Cayley algebra (GCA) [39]. The procedure, omitted for brevity, leads to some geometric conditions for MEPaM's singularities. Considering Fig. 5.5, Right, let $\prod_1$ be the plane passing through $B_1$ and spanned by vectors $\mathbf{u}$ and $\mathbf{z}$.
Let $\prod_3$ be the plane passing through $B_3$ and spanned by vectors $\mathbf{w}$ and $\mathbf{z}$. Let $L_1$ be the intersection line of planes $\prod_1$ and $\prod_3$. Let $L_2$ be the line passing through point $B_2$ and along $\mathbf{v}$. MEPaM reaches a parallel singularity if and only if at least one of the following conditions is verified:

- The four points of the tetrahedron of corners $B_1$, $B_2$, $B_3$, and $\mathbf{z}$ are coplanar, so the moving platform is vertical as in Fig. 5.5, Left
- The lines $L_1$ and $L_2$ intersect as in Fig. 5.5, Right

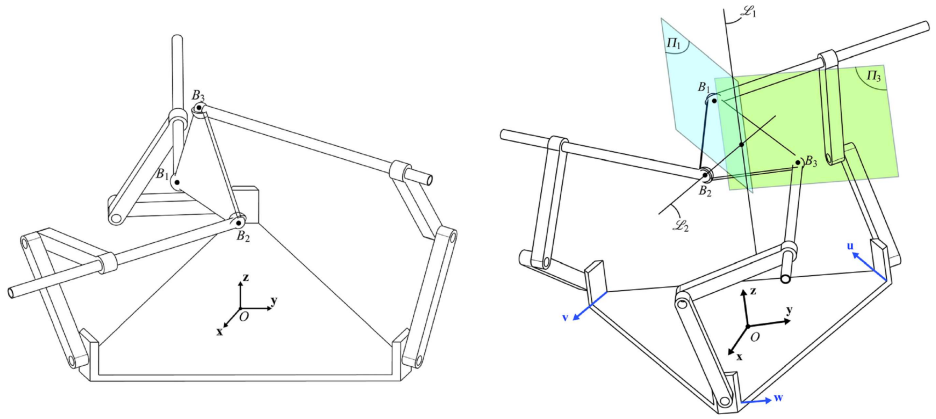



Figure 5.5: Left: A singular configuration of MEPaM where its moving platform is vertical; Right: A singular configuration of MEPaM where lines $L_1$ and $L_2$ intersect

An interesting feature of the manipulator is that the parallel singularity is independent on the position of the end-effector. Moreover, the design of the device prevents the end-effector from reaching the parallel singularities thanks to the physical limits of the universal joints.

## 5.5 Line-based controller singularities: hidden robot model of MEPaM

In this subsection, there is the singularity analysis of the hidden robot model of MEPaM by using Gröbner bases (GB).

Subsection 5.5.1 describes the manipulator under study. Subsection 5.5.2 is devoted to its singularity analysis with Gröbner bases. Finally, the parallel singularities of the manipulator are plotted in its orientation workspace.

### 5.5.1 Description of the hidden robot

Fig. 5.6 shows the hidden robot model of MEPaM. It is composed of an equilateral moving platform connected to the base with three identical legs. Each leg is made by three orthogonal prismatic joints and one spherical joint, the first two prismatic joints being actuated. P stands for a prismatic joint whereas S stands for a spherical joint. An underline letter denotes an actuated joint. Therefore, the manipulator is named 3-PPPS manipulator and provides six-degree-of-freedom motions, i.e., three translations and three rotations. The parametrization follows.

Let $C_1, C_2$ and $C_3$ be the corners of the moving platform (MP) of side length $r$. Let $\mathcal{F}_p$ $(C_p, X_p, Y_p, Z_p)$ be the frame attached to the moving platform, its origin $C_p$ being the centroid of the MP. $Y_p$ is parallel to line $(C_2C_3)$ and $Z_p$ is normal to the MP. Accordingly,

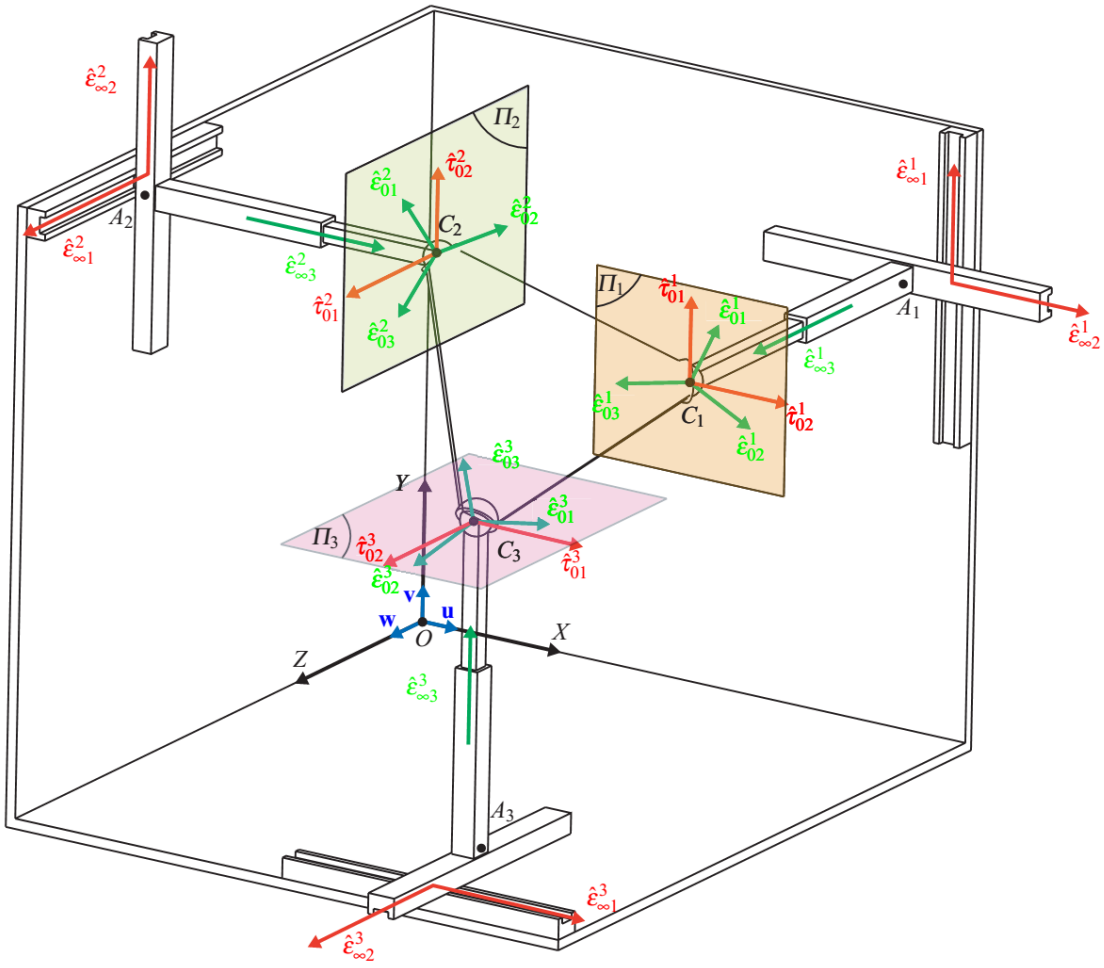



Figure 5.6: The 3-PPPSmanipulator

$$\mathbf{c}_{1p} = \begin{bmatrix} 2r\sqrt{3}/6 \\ 0 \\ 0 \end{bmatrix}, \mathbf{c}_{2p} = \begin{bmatrix} -r\sqrt{3}/6 \\ r/2 \\ 0 \end{bmatrix}, \mathbf{c}_{3p} = \begin{bmatrix} -r\sqrt{3}/6 \\ -r/2 \\ 0 \end{bmatrix} \tag{5.37}$$

are the Cartesian coordinate vectors of points $C_1$, $C_2$ and $C_3$ expressed in $\mathcal{F}_p$. Likewise, let $\mathcal{F}_b$ $(O,X,Y,Z)$ be the frame attached to the base and

$$\mathbf{a}_{1b} = \begin{bmatrix} x_1 \\ y_1 \\ 0 \end{bmatrix}, \mathbf{a}_{2b} = \begin{bmatrix} 0 \\ y_2 \\ z_2 \end{bmatrix}, \mathbf{a}_{3b} = \begin{bmatrix} x_3 \\ 0 \\ z_3 \end{bmatrix} \tag{5.38}$$

be the Cartesian coordinate vectors of points $A_1$, $A_2$ and $A_3$.

### Orientation Space

The orientation space can be fully represented with the variables $(Q_2,Q_3,Q_4)$, a subset of the quaternions coordinates. Indeed, the quaternions represent the rotations of the platform with a rotation axis $\vec{u}$and an angle $\theta$. The relation between the quaternions and the axis and angle

representation can be found in [15]:

$$Q_1 = \cos(\theta/2),\ Q_2 = u_x \sin(\theta/2),\ Q_3 = u_y \sin(\theta/2),\ Q_4 = u_z \sin(\theta/2) \tag{5.39}$$

where $u_x^2 + u_y^2 + u_z^2 = 1$ and $0 \leq \theta \leq \pi$.

Thus, each rotation can be mapped onto a point of the unit ball in the space defined by the variables $(Q_2, Q_3, Q_4)$with the following bijection:

$$\begin{gathered}\Phi : \mathcal{S} \times ]0, \pi] \to \mathcal{B} \setminus \{ \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \} \\ \begin{pmatrix} u_x \\ u_y \\ u_z \end{pmatrix}, \theta \mapsto \begin{pmatrix} Q_2 := u_x \sin(\theta/2) \\ Q_3 := u_y \sin(\theta/2) \\ Q_4 := u_z \sin(\theta/2) \end{pmatrix}\end{gathered} \tag{5.40}$$

where $\mathcal{S}$ is the unit sphere in a 3-dimension space, and $\mathcal{B}$ is the closed unit ball.

When $\theta$is equal to zero, the corresponding rotation matrix is the identity that does not depend on the rotation axis $\vec{u}$. It also maps to the center of $\mathcal{B}$ in the quaternions representation.

**Geometric Model**

Let $\mathbf{c}_b = \begin{bmatrix} c_x & c_y & c_z \end{bmatrix}^T$ be the Cartesian coordinate vector of point $C$, the centroid of the MP, expressed in $\mathcal{F}_b$. The following equations characterize the geometric model of the 3-PPPS manipulator:

$$c_x - 1/3\sqrt{3}{Q_1}^2 - 1/3\sqrt{3}{Q_2}^2 + 1/6\sqrt{3} - Q_2Q_3 + Q_1Q_4 - x_1 = 0 \tag{5.41a}$$

$$c_y - 1/3\sqrt{3}Q_2Q_3 - 1/3\sqrt{3}Q_1Q_4 - {Q_1}^2 - {Q_3}^2 + 1/2 - y_1 = 0 \tag{5.41b}$$

$$c_y - 1/3\sqrt{3}Q_2Q_3 - 1/3\sqrt{3}Q_1Q_4 + {Q_1}^2 + {Q_3}^2 - 1/2 - y_2 = 0 \tag{5.41c}$$

$$c_z - 1/3\sqrt{3}Q_2Q_4 + 1/3\sqrt{3}Q_1Q_3 + Q_3Q_4 + Q_1Q_2 - z_2 = 0 \tag{5.41d}$$

$$c_x + 2/3\sqrt{3}{Q_1}^2 + 2/3\sqrt{3}{Q_2}^2 - 1/3\sqrt{3} - x_3 = 0 \tag{5.41e}$$

$$c_z + 2/3\sqrt{3}Q_2Q_4 - 2/3\sqrt{3}Q_1Q_3 - z_3 = 0 \tag{5.41f}$$

$${Q_1}^2 + {Q_2}^2 + {Q_3}^2 + {Q_4}^2 - 1 = 0 \tag{5.41g}$$

### 5.5.2 Singularity analysis with Gröbner Bases

In this subsection, we focus on the analysis of the parallel singularities of the 3-PPPSmanipulator using the Jacobian and Gröbner bases. We derive the algebraic relations of the singularities satisfied by the orientation variables.

**Jacobian Formulation**

The formula we use to define the parallel singularities is the determinant of a Jacobian matrix. This criterion was introduced in [38], where parallel singularities were referred to singularities of the first type. Equations (5.41a)-(g) depend on six joint variables $\mathbf{T} = (x_1, y_1, y_2, z_2, x_3, z_3)$, six pose variables $(c_x, c_y, c_z, Q_2, Q_3, Q_4)$and one passive variable $(Q_1)$. We denote by $\mathbf{K}$the union of the pose and the passive variables. Let $\mathbf{A}$be the Jacobian matrix of these seven equations with

respect to $\mathbf{K}$, i.e.,

$$\mathbf{A} = \left(\frac{\partial F_i}{\partial \mathbf{K}}\right) =$$

$$\begin{bmatrix}
1 & 0 & 0 & -\frac{2}{3}\sqrt{3}Q_2 - Q_3 & -Q_2 & Q_1 & -\frac{2}{3}\sqrt{3}Q_1 + Q_4 \\
0 & 1 & 0 & -\frac{1}{3}\sqrt{3}Q_3 & -\frac{1}{3}\sqrt{3}Q_2 - 2Q_3 & -\frac{1}{3}\sqrt{3}Q_1 & -\frac{1}{3}\sqrt{3}Q_4 - 2Q_1 \\
0 & 1 & 0 & -\frac{1}{3}\sqrt{3}Q_3 & -\frac{1}{3}\sqrt{3}Q_2 + 2Q_3 & -\frac{1}{3}\sqrt{3}Q_1 & -\frac{1}{3}\sqrt{3}Q_4 + 2Q_1 \\
0 & 0 & 1 & -\frac{1}{3}\sqrt{3}Q_4 + Q_1 & \frac{1}{3}\sqrt{3}Q_1 + Q_4 & -\frac{1}{3}\sqrt{3}Q_2 + Q_3 & \frac{1}{3}\sqrt{3}Q_3 + Q_2 \\
1 & 0 & 0 & \frac{4}{3}\sqrt{3}Q_2 & 0 & 0 & \frac{4}{3}\sqrt{3}Q_1 \\
0 & 0 & 1 & \frac{2}{3}\sqrt{3}Q_4 & -\frac{2}{3}\sqrt{3}Q_1 & \frac{2}{3}\sqrt{3}Q_2 & -\frac{2}{3}\sqrt{3}Q_3 \\
0 & 0 & 0 & 2Q_2 & 2Q_3 & 2Q_4 & 2Q_1
\end{bmatrix} \tag{5.42}$$

Then, let $\mathbf{B}$ be the Jacobian matrix of Eqs. (5.41a)-(g) with respect to the joint variables. It appears that $\mathbf{B}$ is the negative identity matrix. Denoting by $F(\mathbf{K}, \mathbf{T})$ the vector of seven polynomials on the left-hand side of Eqs. (5.41a)-(g), we have:

$$F(\mathbf{K}, \mathbf{T}) = 0 \tag{5.43}$$

Differentiating Eq. (5.43) with respect to time we obtain:

$$\mathbf{A}\dot{\mathbf{K}} - \dot{\mathbf{T}} = 0 \tag{5.44}$$

In particular, we can infer from Eq. (5.44) that parallel singularities occur when the determinant of $\mathbf{A}$vanishes:

$$\begin{aligned}
\det(\mathbf{A}) = {} & -8\,Q_4\sqrt{3}{Q_3}^3 + 48\,{Q_2}^2 Q_3 Q_1 - 48\,Q_2{Q_3}^2 Q_4 \\
& - 24\,\sqrt{3}{Q_2}^2 Q_3 Q_4 + 48\,Q_4 Q_2 {Q_1}^2 + 24\,\sqrt{3}Q_2 Q_1 {Q_4}^2 \\
& + 24\,Q_3\sqrt{3}{Q_1}^2 Q_4 - 48\,Q_3 Q_1 {Q_4}^2 - 24\,Q_2 Q_1\sqrt{3}{Q_3}^2 \\
& + 8\,\sqrt{3}{Q_4}^3 Q_3 - 8\,\sqrt{3}{Q_2}^3 Q_1 + 8\,Q_2\sqrt{3}{Q_1}^3 = 0
\end{aligned} \tag{5.45}$$

Besides, it turns out that the 3-PPPS manipulator does not have any serial singularity as matrix $\mathbf{B}$ is always invertible.

**Singularities in the Workspace**

The singularities of this mechanism can be represented in terms of the pose variables. To this end, we need to eliminate the joint and passive variables from Eq. (5.45) and Eqs. (5.41a)-(g). This can be achieved with methods based on Gröbner basis theory.

Gröbner basis for elimination Let $\mathcal{P}$ be a set of polynomials in the variables $(x_1, y_1, y_2, z_2, x_3, z_3)$and $(c_x,\ c_y,\ c_z,\ Q_2\ ,Q_3,\ Q_4)$. Moreover, let $\mathcal{V}$ be the set of common roots of the polynomials in $\mathcal{P}$, and let $\mathcal{W}$ be the projection of $\mathcal{V}$ on the workspace. It might not be possible to represent $\mathcal{W}$ by polynomial equations. Let $\overline{\mathcal{W}}$ be the smallest set defined by polynomial equations that contains $\mathcal{W}$. Our goal is to compute the polynomial equations defining $\overline{\mathcal{W}}$.

These polynomial equations are computed with Gröbner-basis theory. A Gröbner basis of a polynomial system is a polynomial system equivalent to the first one, and satisfying some additional

specific properties. The Gröbner basis of a system depends on an ordering on the monomials. In our case, if we choose an elimination ordering eliminating $\mathbf{K}$, then the Gröbner basis of $\mathcal{P}$ will contain exactly the polynomials defining $\overline{\mathcal{W}}$.

Equations of the parallel singularities in the workspace We can now use the elimination of the previous paragraph for our problem to obtain the polynomial equations defining implicitly the parallel singularities in the workspace. Let us consider the polynomial set:

$$\begin{cases} F(\mathbf{K},\mathbf{T}) = [0,0,0,0,0,0,0]^T \\ \det(\mathbf{A}) = 0 \end{cases} \tag{5.46}$$

We compute a Gröbner basis of system (5.46) with respect to elimination ordering eliminating $\mathbf{K}$. This computation yields directly the relation satisfied by the parallel singularities in the orientation workspace, namely,

$$\begin{aligned}
&- \; {Q_2}^2 + 9{Q_2}^2{Q_4}^2 + 5{Q_2}^4 + 9{Q_3}^2{Q_2}^2 \\
&+ \; 16{Q_3}^6{Q_2}^2 + 44{Q_3}^4{Q_2}^4 + 32{Q_3}^2{Q_2}^6 - 24{Q_3}^4{Q_2}^2 \\
&- \; 40{Q_3}^2{Q_2}^4 + 16{Q_4}^6{Q_3}^2 + 28{Q_4}^4{Q_3}^4 + 16{Q_4}^2{Q_3}^6 \\
&+ \; 16{Q_4}^6{Q_2}^2 + 48{Q_4}^4{Q_3}^2{Q_2}^2 + 72{Q_4}^2{Q_3}^4{Q_2}^2 + 20{Q_4}^4{Q_2}^4 \\
&+ \; 64{Q_4}^2{Q_3}^2{Q_2}^4 + 8{Q_4}^2{Q_2}^6 - 24{Q_4}^4{Q_3}^2 - 24{Q_4}^2{Q_3}^4 \\
&- \; 48{Q_4}^2{Q_3}^2{Q_2}^2 - 16{Q_4}^2{Q_2}^4 + 9{Q_4}^2{Q_3}^2 + 16{Q_4}^2\sqrt{3}Q_3Q_2 \\
&+ \; 40\sqrt{3}{Q_4}^4{Q_3}^3Q_2 + 48\sqrt{3}{Q_4}^2{Q_3}^5Q_2 + 40\sqrt{3}{Q_4}^4Q_3{Q_2}^3 + 80\sqrt{3}{Q_4}^2{Q_3}^3{Q_2}^3 \\
&+ \; 32\sqrt{3}{Q_4}^2Q_3{Q_2}^5 - 16\sqrt{3}{Q_4}^4Q_3Q_2 - 60\sqrt{3}{Q_4}^2{Q_3}^3Q_2 - 52\sqrt{3}{Q_4}^2Q_3{Q_2}^3 \\
&- \; 8{Q_2}^6 + 4{Q_2}^8 - 4\sqrt{3}{Q_2}^3Q_3 - 16\sqrt{3}{Q_3}^5{Q_2}^3 \\
&- \; 24\sqrt{3}{Q_3}^3{Q_2}^5 - 8\sqrt{3}Q_3{Q_2}^7 + 20\sqrt{3}{Q_3}^3{Q_2}^3 + 12\sqrt{3}Q_3{Q_2}^5 \\
&- \; 24{Q_4}^4{Q_2}^2 = 0
\end{aligned} \tag{5.47}$$

We can notice that these equations depend only on the orientation variables ($Q_2$, $Q_3$, $Q_4$). This means that the parallel singularities do not depend on the position of the centroid of the moving platform. As a matter of fact, the parallel singularities of the 3-PPPS manipulator can be represented in its orientation workspace only, the latter being characterized with variables $(Q_2, Q_3, Q_4)$as shown in Fig. 5.7.

## 5.6 Results

In this section, some examples of simulation results have been done. With the position control scheme presented in section 5.3.3: the end-effector converges correctly from the initial position to the desired position, with the error of the Plücker coordinates of the legs converging to 0. These results have been got with a model in Adams of MEPaM with the following nominal parameters (in meters):
$r_i = 0.165$;
$d_1 = [0.1375, 0.1375, 0.1375]$;
$d_2 = [0.1375, 0.1375, 0.1375]$;
$w_x = [0, 0, 0]$;
$w_z = [0.110, 0.110, 0.110]$;

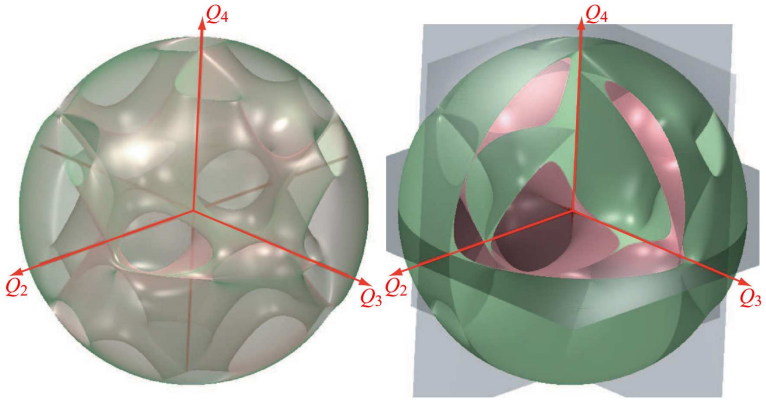
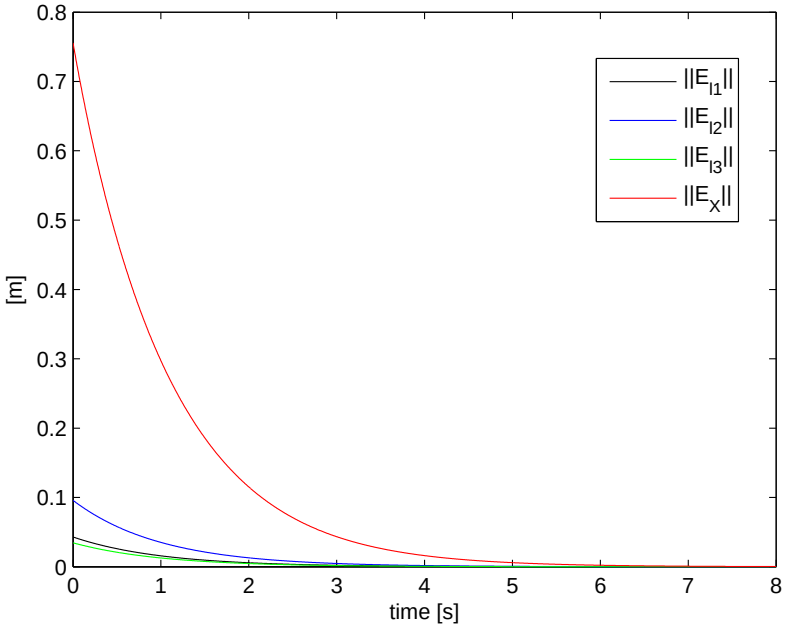



Figure 5.7: The parallel singularities of the 3-PPPSmanipulator in the orientation workspace: given a point M, vector $\overrightarrow{OM}$ defines the orientation axis and its Euclidean norm $\|\overrightarrow{OM}\|$ is the sine of the half-angle of rotation

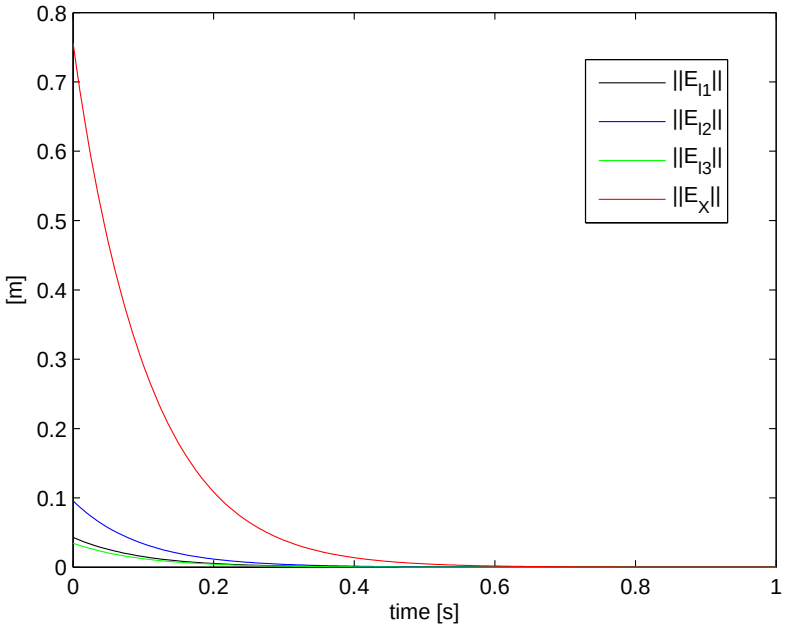
The initial position and orientation of the platform is:
$\mathbf{X}_{d1}$=[0, 0, 0.180, -*pi*/4, 0, 0] (Fig. 5.13(a)).

The desired positions and orientations of the platform are chosen in the workspace, without making the manipulator to cross the singularity of the hidden robot model (which is the same as the singularity of the real robot):
$\mathbf{X}_{d1}$=[-0.051, -0.016, 0.223, -0.577, 0.022, 0.722] (Fig. 5.13(b))
$\mathbf{X}_{d2}$=[0.052, 0.027, 0.174, -0.888, 0.298, 0.554]
$\mathbf{X}_{d3}$=[4.2e-05, 0.049, 0.329, -0.520, 0.271, 0.013]
$\mathbf{X}_{d4}$=[-0.027, 0.045, 0.246, -1.231, -0.200, 0.806]
$\mathbf{X}_{d5}$=[0.043, 0.059, 0.259, -1.096, -0.448, -1.200]

The error converges faster in the case of $\lambda = 10$ than in the case of $\lambda = 1$ (Figs. 5.8, 5.9, 5.10, 5.11, 5.12).

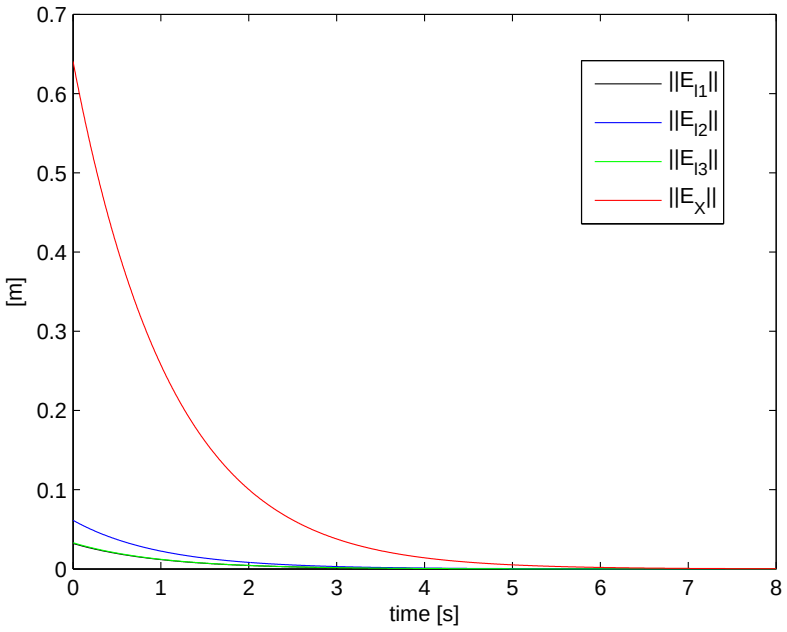



(a) `lambda=1`

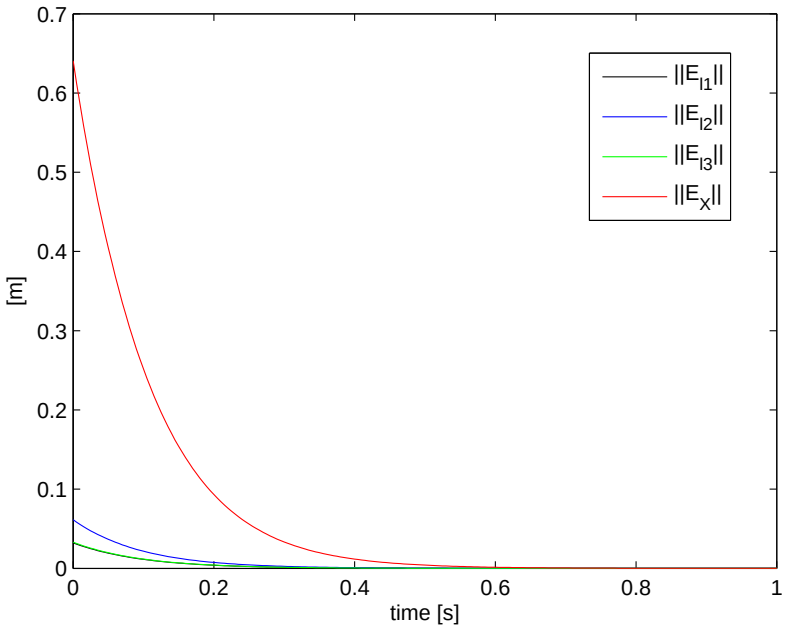



(b) `lambda=10`

Figure 5.8: $\mathbf{X}_{d1}$=[-0.051, -0.016, 0.223, -0.577, 0.022, 0.722]

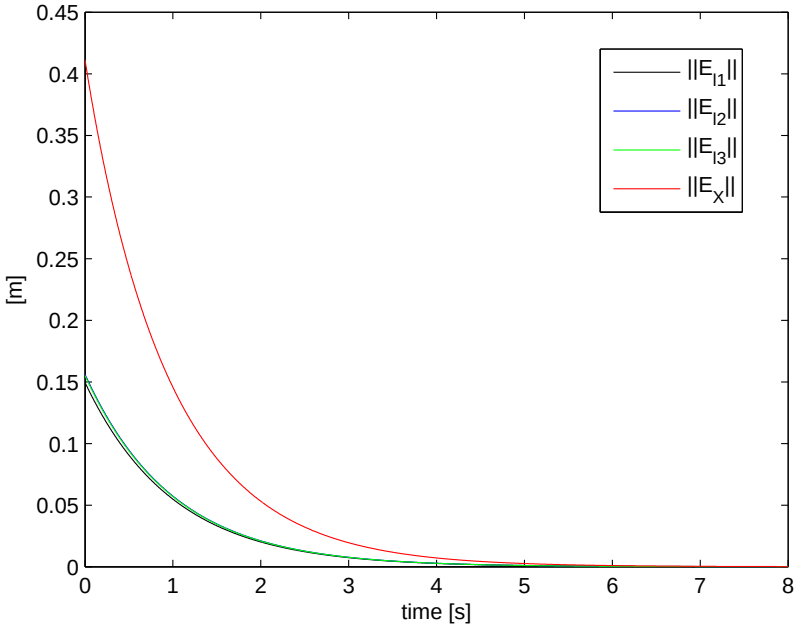



(a) `lambda=1`

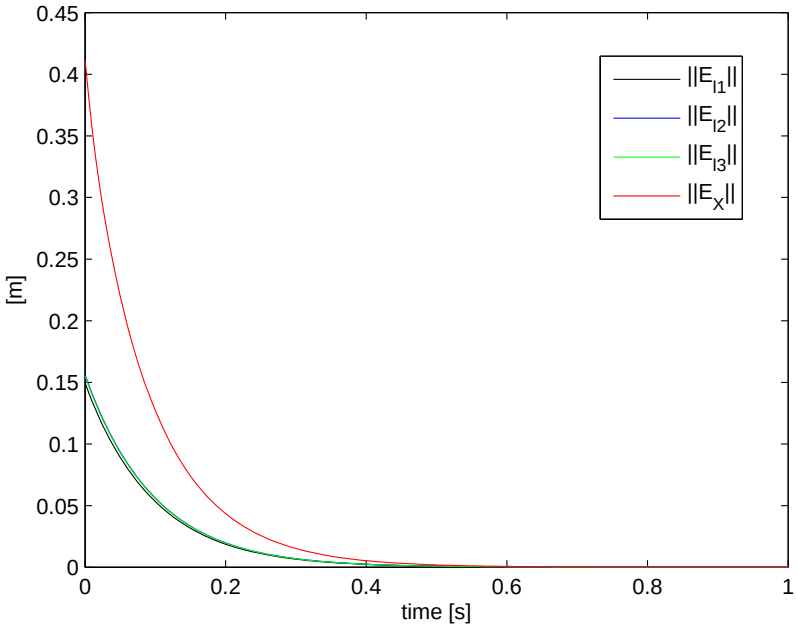



(b) `lambda=10`

Figure 5.9: $\mathbf{X}_{d2}$=[0.052, 0.027, 0.174, -0.888, 0.298, 0.554]

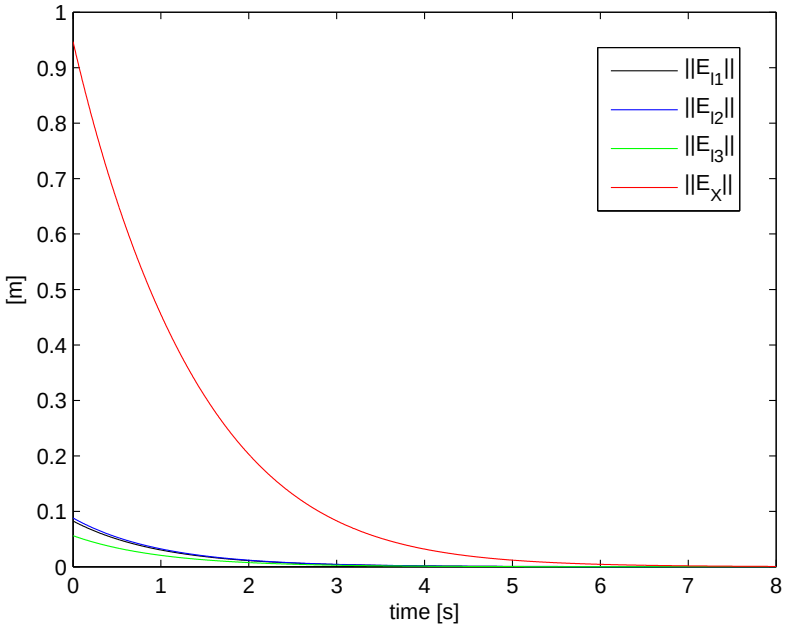



(a) `lambda=1`

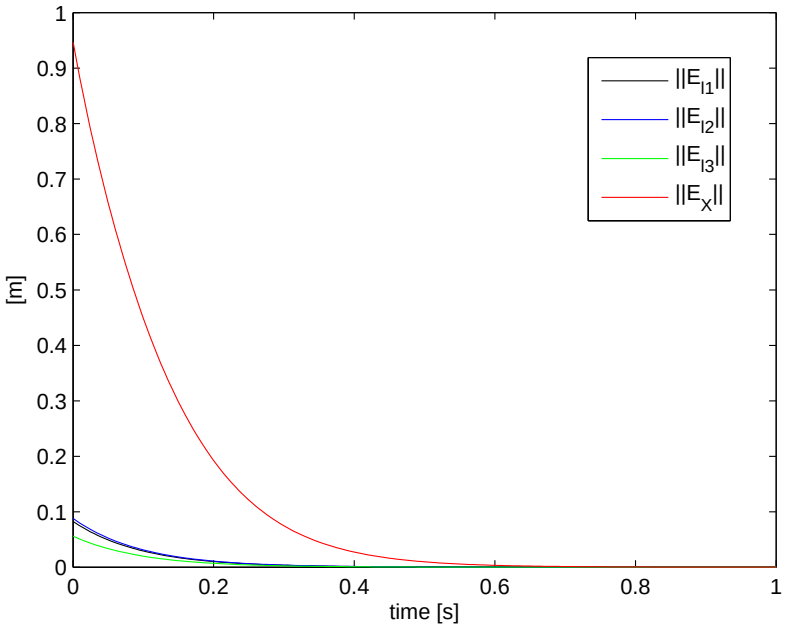



(b) `lambda=10`

Figure 5.10: $\mathbf{X}_{d3}$=[4.2e-05, 0.049, 0.329, -0.520, 0.271, 0.013] ]

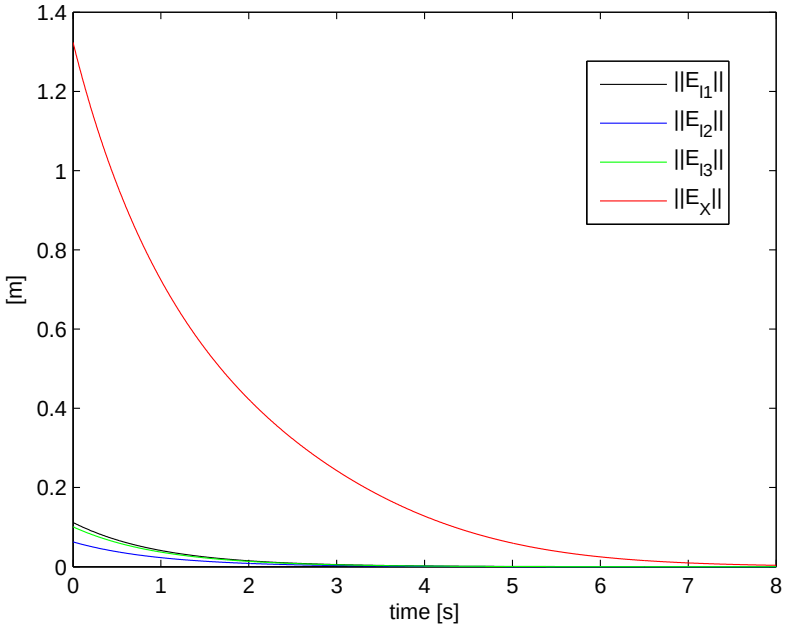



(a) `lambda=1`

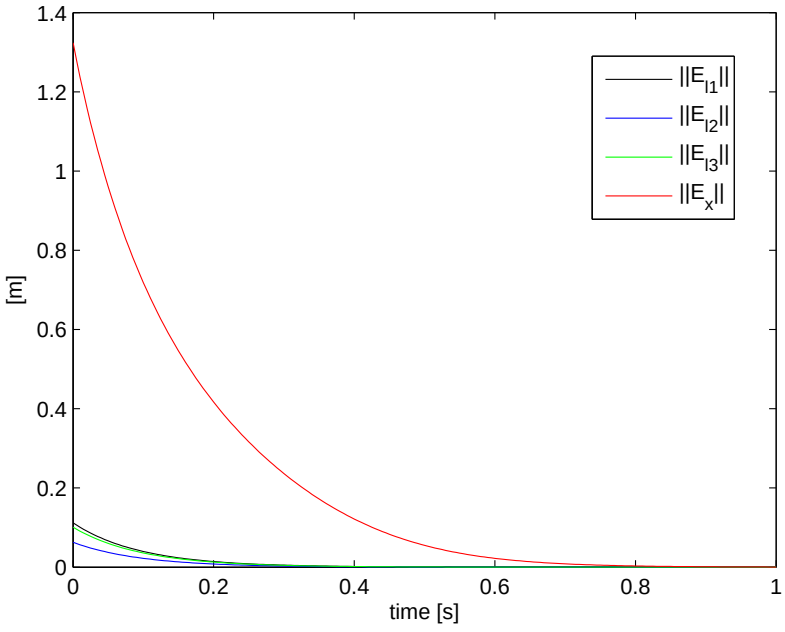



(b) `lambda=10`

Figure 5.11: $\mathbf{X}_{d4}$=[-0.027, 0.045, 0.246, -1.231, -0.200, 0.806]

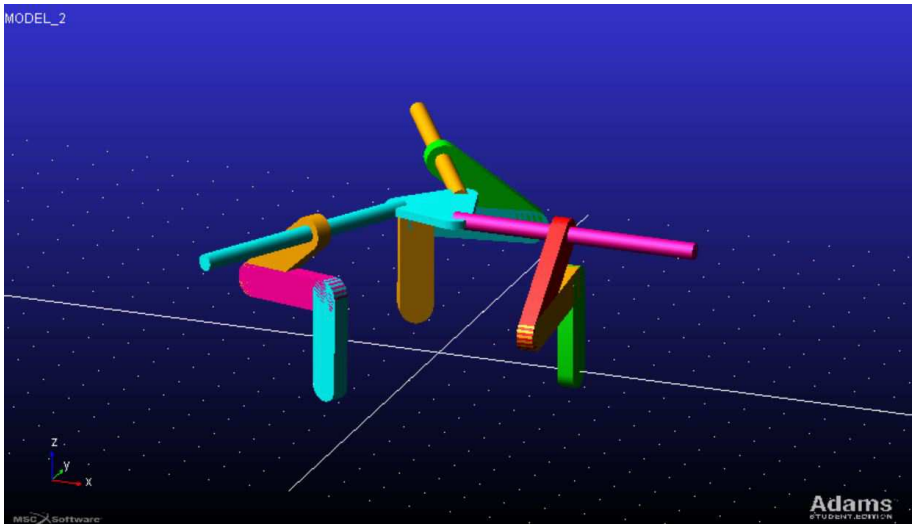



(a) `lambda=1`

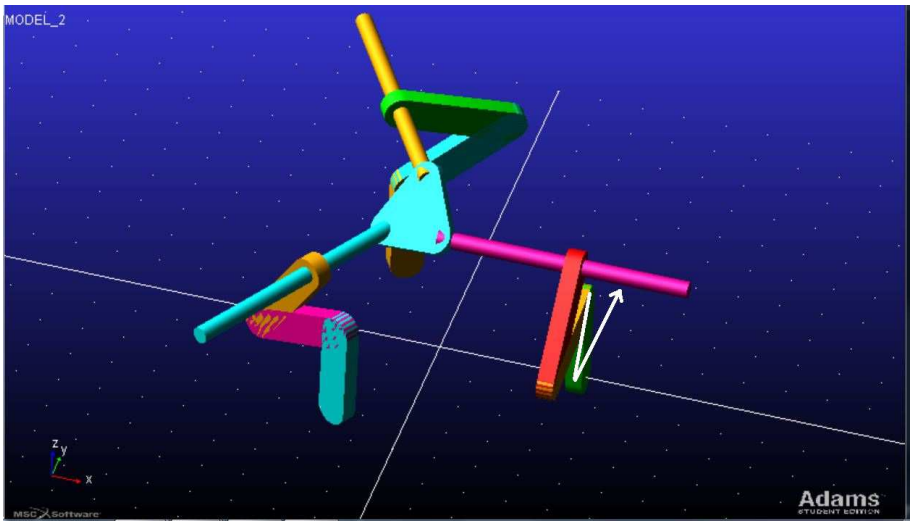



(b) `lambda=10`

Figure 5.12: $\mathbf{X}_{d5}$=[0.043, 0.059, 0.259, -1.096, -0.448, -1.200]




(a) `Initialposition`




(b) `Desiredposition`

Figure 5.13: Initial position $\mathbf{X}_{d0}$ and desired position $\mathbf{X}_{d1}$.

# Chapter 6

# Conclusion and Future Work

## 6.1 Conclusion

This thesis has been in the framework of a collaboration project with the Monash University (Melbourne, Australia).
The aim of this project was to make a position controller of MEPaM, the haptic device based on a parallel mechanism developed at Monash University. For estimating the configuration of the robot and the position of the end-effector, the forward kinematic problem of parallel robots yields multiple solutions. Therefore, it is better to use exteroceptive sensors to estimate the end-effector position.
Past research works show that the robot end-effector pose can be effectively estimated by vision. The most common approach consists of the direct observation of the end-effector pose. However, for haptic devices, the direct observation of the platform may not be easy due to the presence of the user hand while it is generally not a problem to observe its legs that are most often designed with slim and rectilinear rods.
A first step in this direction was made by some researchers of the IRCCyN Robotics Team where vision was used to derive a visual servoing scheme based on the observation of the legs directions. Nevertheless, this approach has two major drawbacks:

- it is not suitable for some PKM families (e.g. MEPaM, whose legs directions are constant even if the end-effectors pose changes),
- it involves the presence of some models of robots, different from the real one, "'hidden"' into the controller (named "'hidden robot model"').

For overcoming both these problems, in this thesis we have proposed another visual servoing approach, based on the extraction of the Plücker coordinates of the legs lines. The new line-based approach has been applied to a five-bar mechanism and to MEPaM.
Being the five-bar mechanism suitable for both the approaches, it has been possible to develop both the controllers using Simulink and to do a comparative analysis: it has been proven that the new approach is the best one because its hidden robot model has the same singularities as the real robot, therefore it is much more robust to the measurement noise near the singularities of the hidden robot of the old approach and it can also pass through them.
Then, the new approach has been applied to MEPaM: some results of simulations have been shown and its hidden robot model has been analyzed.

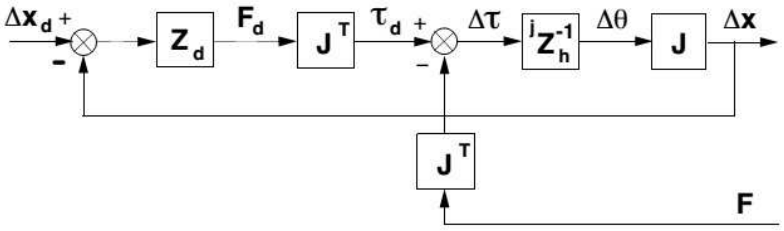



Figure 6.1: The open-loop force control.
Symbols: virtual environment impedance $\mathbf{Z}_{env}$, penetration of the avatar $\mathbf{x}$ through the virtual environment $\mathbf{x}_{env}$, desired force $\mathbf{F}_d$, admittance $\mathbf{Z}_h^{-1}$, inverse Jacobian $\mathbf{J}^T$ which maps $\mathbf{F}_d$ into desired torque commands $\tau_d$, which are applied to the haptic device.

## 6.2 Future Work

Throughout the course of this thesis, some new ideas came out for the future development of the project.

One idea is to develop a new control scheme based just on the interaction matrix but not on the Jacobian matrix: in that way, the controller will be completely independent to the geometric parameters of the robot like the length of the links. Therefore, in a real environment, the noise due to the error on the geometric parameters will not be relevant at all.

Another hint could be to implement the controller using an object-oriented approach for speeding up the computations in Matlab.

Another idea is to develop a visual-based force control for MEPaM. Some researcher at the Monash university have already develop an impedance force controller (Fig 6.1): to turn it into a vision-based force controller, it will be necessary to arrange at least one camera for observing the legs of the robot and to calibrate it and the block for solving the direct kinematic problems should be replaced with the one based on the Plücker coordinates described in Section 5.3.2.

# Bibliography


[1] T. Leinonen. Terminology for the theory of machines and mechanisms. *Mechanism and Machine Theory*, 26, 1991.

[2] J.P. Merlet. *Parallel Robots*. Springer, 2nd edition, 2006.

[3] J.P. Merlet. www-sop.inria.fr/members/jean-pierre.merlet/merlet.html. 2012.

[4] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3), 1992.

[5] R. Horaud, F. Dornaika, and B. Espiau. Visually guided object grasping. *IEEE Transactions on Robotics and Automation*, 14(4):525–532, 1998.

[6] P. Martinet, J. Gallice, and D. Khadraoui. Vision based control law using 3D visual features. In *Proceedings of the World Automation Congress, WAC96, Robotics and Manufacturing Systems*, volume 3, pages 497–502, Montpellier, France, May 1996.

[7] N. Andreff, A. Marchadier, and P. Martinet. Vision-based control of a Gough-Stewart parallel mechanism using legs observation. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'05*, pages 2546–2551, Barcelona, Spain, April 18-22 2005.

[8] V.E. Gough and S.G. Whitehall. Universal tyre test machine. In *Proceedings of the FISITA 9th International Technical Congress*, pages 117–317, May 1962.

[9] E. Ozgur, N. Andreff, and P. Martinet. Dynamic control of the quattro robot by the leg edgels. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA11*, Shanghai, China, May 9-13 2011.

[10] N. Andreff and P. Martinet. Vision-based kinematic modelling of some parallel manipulators for control purposes. In *Proceedings of EuCoMeS, the First European Conference on Mechanism Science*, Obergurgl, Austria, 2006.

[11] V. Rosenzveig, S. Briot, P. Martinet, E. Ozgur, and N. Bouton. A method for simplifying the analysis of leg-based visual servoing of parallel robots. In *Proc. 2014 IEEE Int. Conf. on Robotics and Automation (ICRA 2014)*, Hong Kong, China, May 2014.

[12] S. Briot and P. Martinet. Minimal representation for the control of Gough-Stewart platforms via leg observation considering a hidden robot model. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA 2013)*, Karlsruhe, Germany, May, 6-10 2013.

[13] V. Rosenzveig, S. Briot, and P. Martinet. Minimal representation for the control of the Adept Quattro with rigid platform via leg observation considering a hidden robot model. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013)*, Tokyo Big Sight, Japan, 2013.

[14] N. Andreff F. Paccot and P. Martinet. *A Review on the Dynamic Control of Parallel Kinematic Machines: Theory and Experiments*. The International Journal of Robotics Research, 2009.

[15] W. Khalil and E. Dombre. *Modeling, Identification and Control of Robots*. 2002.

[16] J.P . Merlet. *Parallel robots*. Kluwer Academic Publishers, 2000.

[17] J. Gallice P . Martinet and D. Khadraoui. *Vision based control law using 3d visual features*. Proc. World Automation Congress, WAC'96, Robotics and Manufacturing Systems, May 1996.

[18] F . Chaumette B. Espiau and P . Rives. *A New Approach To Visual Servoing in Robotics*. IEEE Trans. on Robotics and Automation, June 1992.

[19] A. Marchadier N. Andreff and P. Martinet. *Vision-based control of a Gough-Stewart parallel mechanism using legs observation*. Proceedings of the IEEE International Conference on Robotics and Automation, April 2005.

[20] T. Dallej N. Andreff and P. Martinet. *Image-based visual servoing of gough-stewart parallel manipulators using legs observation*. International Journal of Robotics Research, 2007.

[21] N. Andreff and P. Martinet. *Vision-based kinematic modelling of some parallel manipulators for control purposes*. Proceedings of EuCoMeS, the First European Conference on Mechanism Science, 2006.

[22] N. Andreff E. Ozgur and P. Martinet. *Dynamic control of the quattro robot by the leg edgels*. Proceedings of the IEEE International Conference on Robotics and Automation, May 2011.

[23] S. Briot and P. Martinet. *Minimal representation for the control of Gough-Stewart platforms via leg observation considering a hidden robot model*. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, 2013.

[24] N. Andreff, B. Espiau, and R. Horaud. Visual servoing from lines. *International Journal of Robotics Research*, 21(8):679–700, 2002.

[25] J. Plücker. On a new geometry of space. *Philosophical Transactions of the Royal Society of London*, 155:725–791, 1865.

[26] N. Andreff, T. Dallej, and P. Martinet. Image-based visual servoing of gough-stewart parallel manipulators using legs observation. *International Journal of Robotics Research*, 26(7):677–687, 2007.

[27] F. Chaumette. *La commande des robots manipulateurs*. Hermès, 2002.

[28] J.M. Selig. Geometric fundamentals of robotics, 2005. Springer, 2nd edition.

[29] C.M. Gosselin and J. Angeles. Singularity analysis of closed-loop kinematic chains. *IEEE Transactions on Robotics and Automation*, 6(3):281–290, 1990.

[30] F. Behi. *Kinematic Analysis for a Six-Degree-of-Freedom 3-PRPS Parallel Mechanism*. IEEE J. Rob. Innov., 1988.

[31] Lee S. Tsai K. Kohli, D. and G. Sandor. *Manipulator Configurations Based on Rotary-Linear (r-l) Actuators and Their Direct and Inverse Kinematics*. J. Mech., Transm., Autom. Des., 1988.

[32] K. Cleary and T. Brooks. *Kinematic Analysis of a Novel 6-DOF Parallel Manipulator*. IEEE International Conference on Robotics and Automation, 1993.

[33] Y. Byun and H. S. Cho. *Analysis of a Novel 6-DOF, 3-PPSP Parallel Manipulator*. Int. J. Rob. Res., 1997.

[34] Glozman D. Simaan, N. and M. Shoham. *Design Considerations of New Six Degrees-of-Freedom Parallel Robots*. IEEE International Conference on Robotics and Automation, 1998.

[35] S.-U. Lee and S. Kim. *Analysis and Optimal Design of a New 6 DOF Parallel Type Haptic Device*. IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006.

[36] B. Monsarrat and C. Gosselin. *Workspace Analysis and Optimal Design of a 3-Leg 6-DOF Parallel Platform Mechanism*. IEEE Trans. Rob.Autom., 2003.

[37] S. Carol D. Chablat G. Moroz S. Abeywardena C. Chen, T. Gayral. *A Six Degree of Freedom Epicyclic-Parallel Manipulator*. 2012.

[38] C. Gosselin and J. Angeles. *Singularity Analysis of Closed-Loop Kinematic Chains*. IEEE Trans. Rob. Autom., 1990.

[39] Moroz G. Gayral T. Chablat D. Caro, S. and C. Chen. *Singularity Analysis of a Six-DOF Parallel Manipulator Using Grassmann-Cayley Algebra and Grobner Bases*. Symposium on Brain, Body and Machine, Montreal, QC.,Canada, 2010.