

# Thèse de Doctorat

**Victor ROSENZVEIG**

*Mémoire présenté en vue de l'obtention du  
**grade de Docteur de l'École centrale de Nantes**  
sous le label de l'Université de Nantes Angers Le Mans*

**École doctorale : Sciences et technologies de l'information, et mathématiques**

**Discipline : Automatique, productique et robotique**

**Unité de recherche : Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN)**

**Soutenue le 25 septembre 2015**

## Sensor-Based Design and Control of High-Speed Manipulators

### JURY

Rapporteurs : **M. Nicolas ANDREFF**, Professeur des universités, Institut FEMTO-ST  
**M. Jean-Pierre MERLET**, Directeur de recherche, INRIA Sophia Antipolis

Examineurs : **M. Nicolas BOUTON**, Maître de conférences, Institut Pascal  
**M. François CHAUMETTE**, Directeur de recherche, INRIA Rennes

Directeur de thèse : **M. Philippe MARTINET**, Professeur des universités, École Centrale de Nantes

Co-directeur de thèse : **M. Sébastien BRIOT**, Chargé de recherche, IRCCyN



# Summary

It is well known that parallel manipulators have better dynamic performance than their serial counterparts. However, they have very complex models and the classical control schemes used are susceptible to modelling errors. In order to suppress these errors, it is possible to use exteroceptive sensors to measure the end-effector pose directly.

Cameras can offer this opportunity, providing higher accuracy than in the case of model-based control schemes. In some cases, it is impossible to directly observe the end-effector, and the robot leg directions can be observed instead. However, this leads to unusual problems of non-convergence, presented in the first part of this manuscript. These results can be explained through the use of the hidden robot concept, which is a tangible visualisation of the mapping between the observed leg direction space and the Cartesian space.

The second part of this manuscript offers a formalisation and generalisation of the hidden robot concept. It is then applied to an Adept Quattro, through simulations and experiments, proving its usefulness in terms of convergence and accuracy analysis, as well as finding singularities and local minima within the mapping.

In the third part, further applications based on the hidden robot concept are developed. This allows for a controllability analysis of different planar and spatial robot families using simple geometrical methods. Then, the hidden robot concept is used for analyses which provide algorithms for selecting the optimal set of legs to be observed, as well as providing means for optimal feature selection.

Finally, in the fourth part, the hidden robot concept is used as a tool for sensor-based design, by integrating the needs of the sensor-based control scheme into the design process, specifically for optimisation of the geometric parameters of a Five-Bar mechanism. This design is then compared to one which was obtained through classical means, ignoring the hidden robot concept.





# Résumé

Il est bien connu que les manipulateurs parallèles ont de meilleures performances dynamiques que leurs homologues sériels. Cependant, ils ont des modèles très complexes et les systèmes de contrôle classiques utilisés sont sensibles aux erreurs de modélisation. Afin de supprimer ces erreurs, il est possible d'utiliser des capteurs extéroceptifs pour mesurer directement la pose de l'effecteur.

Les caméras peuvent être utilisées, offrant une précision plus élevée que dans le cas de contrôleurs à base de modèles. Dans certains cas, il est impossible d'observer directement l'effecteur, et les directions des jambes du robot peuvent être observées à la place. Toutefois, cela conduit à des problèmes inhabituels de non-convergence, présentés dans la première partie de ce manuscrit. Ces résultats peuvent être expliqués par l'utilisation du concept de robot caché, qui est une visualisation concrète de la cartographie entre l'espace des directions des jambes observées et l'espace cartésien.

La deuxième partie de ce manuscrit présente la formalisation et la généralisation du concept de robot caché. Ce concept est ensuite validé sur un Adept Quattro, par des simulations et des expériences, ce qui prouve son utilité en termes d'analyse de la convergence et de la précision, ainsi que pour trouver les singularités et les minima locaux dans la cartographie.

Dans la troisième partie, de nouvelles applications basées sur le concept de robot caché sont développées. Cela permet une analyse de contrôlabilité pour différentes familles de robots plans et spatiaux en utilisant des méthodes géométriques simples. Ensuite, le concept de robot caché est utilisé pour fournir des algorithmes de sélection de l'ensemble optimal de jambes à observer.

Enfin, dans la quatrième partie, le concept de robot caché est utilisé pour la conception basée capteurs, en intégrant les besoins du système de contrôle basé capteurs dans le processus de conception, en particulier pour l'optimisation des paramètres géométriques d'un mécanisme à cinq barres. Cette conception est ensuite comparée à celle qui a été obtenue par des moyens classiques, en ignorant le concept de robot caché.



# Contents

<b>Summary</b>	<b>i</b>
<b>Résumé</b>	<b>iii</b>
<b>Contents</b>	<b>v</b>
<b>List of Figures</b>	<b>ix</b>
<b>List of Tables</b>	<b>xiii</b>
<b>List of Symbols</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 State of the Art in Parallel Robots and Vision-Based Control</b>	<b>5</b>
2.1 Parallel Robots . . . . .	5
2.1.1 What Is a Robot? . . . . .	5
2.1.2 Modelling Parallel Robots . . . . .	9
2.2 Classic Control Methods . . . . .	14
2.2.1 PID Control . . . . .	14
2.2.2 Computed Torque Control . . . . .	15
2.2.3 Classical Control Comparison . . . . .	16
2.3 Vision-Based Control . . . . .	17
2.3.1 Why Use Vision? . . . . .	17
2.3.2 Basics of Vision-Based Control . . . . .	18
2.3.3 Vision-Based Control of the GS Platform . . . . .	25
2.4 Conclusions . . . . .	27
<b>3 The Hidden Robot Concept</b>	<b>29</b>
3.1 Preamble Regarding the Problems Raised by Vision-Based Control . . . . .	29
3.1.1 Description of the GS Platform’s Hidden Robot Architecture . . . . .	29
3.1.2 Accuracy Analysis . . . . .	32
3.1.3 Simulation Results . . . . .	33
3.2 The Hidden Robot Concept . . . . .	35

3.2.1	The Philosophy Behind the Hidden Robot . . . . .	35
3.2.2	How to Obtain the Hidden Robot Architecture . . . . .	35
3.2.3	Answers Provided by the Hidden Robot . . . . .	39
3.3	Application of the Hidden Robot to the Adept Quattro . . . . .	41
3.3.1	Kinematics of the Quattro . . . . .	41
3.3.2	Simulation Results . . . . .	47
3.3.3	Experimental Results . . . . .	52
3.4	Conclusions . . . . .	56
<b>4</b>	<b>Extending the Applications of the Hidden Robot Concept</b>	<b>59</b>
4.1	Analysis of the Hidden Robots of Planar Manipulators . . . . .	59
4.1.1	The Hidden Robot Legs of Planar Parallel Robots . . . . .	59
4.1.2	The Hidden Robot Models of Planar Parallel Robots . . . . .	61
4.2	Analysis of the Hidden Robots of Spatial Manipulators . . . . .	64
4.2.1	The $n$ -Pod Robot Family . . . . .	64
4.2.2	The Delta-like Robot Family . . . . .	64
4.3	Optimal Leg Selection . . . . .	67
4.3.1	Questions Regarding Leg Selection . . . . .	67
4.3.2	Leg Selection Algorithms . . . . .	68
4.3.3	Application of the Leg Selection Algorithm to the Adept Quattro . . . . .	70
4.4	Controllability Analysis Using the Hidden Robot . . . . .	72
4.4.1	Robots Which Are Not Controllable Using the Leg Direction Observation	72
4.4.2	Robots Which Are Partially Controllable in Their Whole Workspace Using the Leg Direction Observation . . . . .	73
4.4.3	Robots Which Are Fully Controllable in Their Whole Workspace Using the Leg Direction Observation . . . . .	74
4.5	Proper Feature Observation . . . . .	75
4.5.1	Illustrative Example . . . . .	75
4.5.2	Robots Which Become Fully Controllable in Their Whole Workspace if Additional Information Is Used . . . . .	80
4.6	Conclusions . . . . .	81
<b>5</b>	<b>The Hidden Robot as a Sensor-Based Design Tool</b>	<b>83</b>
5.1	Classical Design Methodology . . . . .	83
5.1.1	Classical Design Approach . . . . .	83
5.1.2	Classical Design Optimisation . . . . .	84
5.2	Optimal Design of the Five-Bar Using the Classical Approach . . . . .	85
5.3	Sensor-Based Design Methodology . . . . .	88
5.4	Optimal Design of the Five-Bar Using the Sensor-Based Approach . . . . .	89
5.5	Design Comparison . . . . .	92
5.6	Conclusions . . . . .	94

---

<b>6</b>	<b>Conclusions</b>	<b>97</b>
6.1	Conclusions . . . . .	97
6.2	Research Perspectives . . . . .	98
<b>A</b>	<b>Assembly modes of the studied <i>ppm</i> and of their hidden robots</b>	<b>101</b>
	<b>Bibliography</b>	<b>109</b>



# List of Figures

2.1	Serial robot . . . . .	6
2.2	Baxter robot . . . . .	7
2.3	Stewart platform (parallel robot) . . . . .	8
2.4	Quattro robot . . . . .	8
2.5	Planar <u>RRRP</u> mechanism . . . . .	12
2.6	Planar <u>RRRP</u> mechanism in a type 1 singularity . . . . .	12
2.7	Planar <u>RRRP</u> mechanism in a type 2 singularity . . . . .	13
2.8	Planar <u>RRRP</u> mechanism in a type 3 singularity . . . . .	13
2.9	PID control scheme . . . . .	15
2.10	Computed torque control scheme . . . . .	16
2.11	A general model-based control scheme . . . . .	17
2.12	A general sensor-based control scheme . . . . .	18
2.13	Projection of a cylinder in the image . . . . .	23
2.14	A GS platform from DeltaLab. . . . .	26
2.15	Duality between the mobile end-effector mode and the fixed end-effector mode. . . . .	26
3.1	Solutions of the <i>fkp</i> for a 3- <u>UPS</u> robot . . . . .	30
3.2	A 3- <u>UPS</u> robot. . . . .	31
3.3	Maximal pose error (in mm) for $z = 0.4$ m when the platform is at zero orientation. . . . .	33
3.4	Error on each leg $e_i^T e_i$ . . . . .	34
3.5	Trajectory in space with initial, desired and final platform positions. . . . .	34
3.6	Simulated platform pose error (in mm). . . . .	35
3.7	A general robot leg and its corresponding hidden robot leg . . . . .	36
3.8	Parametrisation of a unit vector with respect to a given frame . . . . .	36
3.9	A <u>RU</u> leg and two equivalent solutions for its hidden leg . . . . .	38
3.10	Two configurations of a five bar mechanism with identical leg directions . . . . .	40
3.11	Example of leg and of robot of the Delta-like family . . . . .	41
3.12	The platform of the Quattro . . . . .	41
3.13	Solutions of the <i>fkp</i> for a 2- $\Pi$ -{2- <u>UU</u> } robot (in this example, only 4 assembly modes exist) . . . . .	45
3.14	Example of a Type 2 singularity for a 2- $\Pi$ (2- <u>UU</u> ) robot: the platform gets an uncontrollable translation. . . . .	46

3.15	Maximal position error (in mm) for $z = -0.7$ m and $\phi = 0$ deg. . . . .	46
3.16	Maximal orientation error (in deg) for $z = -0.7$ m and $\phi = 0$ deg. . . . .	46
3.17	Maximal position and orientation error for $z = -0.61$ m and $\phi = 0$ deg when all legs are observed. . . . .	47
3.18	The ADAMS mockup of the Adept Quattro connected to Matlab/Simulink via the module ADAMS/Controls . . . . .	47
3.19	The controller used for the simulations . . . . .	48
3.20	Error norm on each leg $\ e_i\ $ . . . . .	49
3.21	Initial, desired and final position when legs $\{1, 4\}$ are observed . . . . .	49
3.22	Error norm on each leg $\ e_i\ $ when all the legs are observed. . . . .	49
3.23	Initial, desired and final position when all legs are observed . . . . .	50
3.24	Platform velocity and error norm on each leg when the robot meets a local minimum. . . . .	50
3.25	Robot configuration when it meets a local minimum. . . . .	51
3.26	Orientation and position error when legs $\{2, 3\}$ and legs $\{2, 4\}$ are observed. . .	51
3.27	Orientation and position error when legs $\{1, 4\}$ , $\{1, 3, 4\}$ and all legs are observed. .	52
3.28	Experimental bench . . . . .	53
3.29	The Quattro recovered by the cloth . . . . .	53
3.30	Convergence of the robot when legs 1 and 4 are observed (desired pose: $\{x = -0.2, y = 0, z = -0.56, \phi = 0\}$ ). . . . .	54
3.31	Convergence of the robot when legs 2 and 3 are observed (desired pose: $\{x = -0.2, y = 0, z = -0.56, \phi = 0\}$ ). . . . .	55
3.32	Convergence of the robot when all legs are observed (desired pose: $\{x = 0.03, y = 0.03, z = -0.59, \phi = 0\}$ ). . . . .	55
4.1	The 3- <u>RRR</u> robot and its hidden robot model . . . . .	62
4.2	Solutions of the <i>fkp</i> for a 3- <u>IRR</u> robot . . . . .	63
4.3	Example of Type 2 singularities for the 3- <u>IRR</u> robot . . . . .	63
4.4	A 2- <u>II(2-UU)</u> robot. . . . .	66
4.5	Motion of the GS Platform using legs 1,3,5 . . . . .	69
4.6	Motion of the GS Platform using legs 1,2,5 (after leg selection algorithm) . . .	69
4.7	Errors in position and leg directions while using online leg selection . . . . .	70
4.8	Errors in position and orientation while using online leg selection . . . . .	71
4.9	The <u>PRRR</u> robot and its hidden robot model . . . . .	73
4.10	The Orthoglide and its hidden robot leg. . . . .	74
4.11	Schematics of the 3- <u>PRR</u> robot. . . . .	76
4.12	Singularities of the 3- <u>PRR</u> robot. . . . .	77
4.13	Hidden robots involved in the tested visual servoings of the 3- <u>PRR</u> robot. . . .	78
4.14	The hidden robot leg when the Plücker coordinates of the line passing through the axis of the leg are observed. . . . .	81



---

5.1	Typical French design methodology . . . . .	84
5.2	The planar footprint of the five-bar in its home position . . . . .	86
5.3	Transmission factor . . . . .	86
5.4	Largest regular dexterous workspace and hidden robot singularities for the camera- optimised architecture . . . . .	88
5.5	Integration of the hidden robot concept into the design process . . . . .	88
5.6	Hidden robot corresponding to the five-bar mechanism . . . . .	89
5.7	Result of a 1 pixel error on the boundary intersect of the observed line . . . . .	90
5.8	Influence of the leg radius on the end-effector error resolution . . . . .	91
5.9	Largest regular dexterous workspace and hidden robot singularities for the camera- optimised architecture . . . . .	92
5.10	Simulations using the encoder-optimised design. . . . .	93
5.11	Simulations using the camera-optimised design. . . . .	93
5.12	Positioning error at point $\{x = -0.15m, y = -0.2m\}$ of the encoder-optimised design . . . . .	94
5.13	Positioning error at point $\{x = -0.15m, y = -0.2m\}$ of the camera-optimised design . . . . .	94



# List of Tables

3.1	Geometric characteristics of the Adept Quattro . . . . .	42
3.2	Results on the experiments carried out for testing the convergence of the robot when legs 1 and 4 are observed (the positions are in meters, the angles in radians). . . . .	54
3.3	Results on the experiments carried out for testing the convergence of the robot when legs 2 and 3 are observed (the positions are in meters, the angles in radians). . . . .	55
3.4	Results on the experiments carried out for testing the convergence of the robot all legs are observed (the positions are in meters, the angles in radians). . . . .	56
3.5	Results on the experiments carried out for testing the accuracy of the robot when legs $\{2, 3\}$ or $\{2, 4\}$ are observed. . . . .	56
3.6	Results on the experiments carried out for testing the accuracy of the robot when legs $\{1, 4\}$ , $\{1, 3, 4\}$ or $\{1, 2, 3, 4\}$ are observed. . . . .	56
4.1	The 10 possible architectures for the legs of <i>ppm</i> and their equivalent hidden robot leg for the visual servoing using leg directions . . . . .	60
4.1	The 10 possible architectures for the legs of <i>ppm</i> and their equivalent hidden robot leg for the visual servoing using leg directions . . . . .	61
4.2	Final end-effector configuration for the desired end-effector configuration $x_f = 0.20$ m, $y_f = 1.03$ m and $\phi_f = -10$ deg . . . . .	79
4.3	Final end-effector configuration for the desired end-effector configuration $x_f = 0.20$ m, $y_f = 1.03$ m and $\phi_f = +10$ deg . . . . .	79
5.1	Specifications for the robot . . . . .	85
5.2	Datasheet of the ETEL TMB140-70 motor . . . . .	85
5.3	Datasheet of the Mikrotron EoSens 4CXP CoaXPress camera with Kowa LM12XC lens . . . . .	85
5.4	Results of the classical optimisation . . . . .	87
5.5	Results of the vision-based optimisation . . . . .	91
A.1	Architectures of the 6 usual <i>ppm</i> with 2 <i>dof</i> , their assembly modes and their hidden robot models for the visual servoing using leg directions . . . . .	101
A.1	Architectures of the 6 usual <i>ppm</i> with 2 <i>dof</i> , their assembly modes and their hidden robot models for the visual servoing using leg directions . . . . .	102

---

A.1	Architectures of the 6 usual <i>ppm</i> with 2 <i>dof</i> , their assembly modes and their hidden robot models for the visual servoing using leg directions . . . . .	103
A.1	Architectures of the 6 usual <i>ppm</i> with 2 <i>dof</i> , their assembly modes and their hidden robot models for the visual servoing using leg directions . . . . .	104
A.2	Architectures of the 6 usual <i>ppm</i> with 3 <i>dof</i> , their number of assembly modes (outside of singular configurations) and their hidden robot models for the visual servoing using leg directions . . . . .	105
A.2	Architectures of the 6 usual <i>ppm</i> with 3 <i>dof</i> , their number of assembly modes (outside of singular configurations) and their hidden robot models for the visual servoing using leg directions . . . . .	106
A.2	Architectures of the 6 usual <i>ppm</i> with 3 <i>dof</i> , their number of assembly modes (outside of singular configurations) and their hidden robot models for the visual servoing using leg directions . . . . .	107

# List of Symbols

This nomenclature references the principal variables and abbreviations used in this manuscript. Unless otherwise noted, the following conventions are used:

- vectors and matrices in **bold style**
- scalar variables and names of points in *italic style*
- axes names in ***bold italic style***
- abbreviations of terms in regular style

$\mathbf{R}, \underline{\mathbf{R}}$	a passive/active revolute joint
$\mathbf{T}, \underline{\mathbf{T}}$	a passive/active translational joint
$\mathbf{U}, \underline{\mathbf{U}}$	a passive/active universal joint
$\mathbf{S}, \underline{\mathbf{S}}$	a passive/active spherical joint
$\Pi$	a passive planar parallelogram joint
$i$	an integer; $i = 1, 2, \dots$
$j$	an integer; $j = 1, 2, \dots$
$\mathbf{x}$	the vector of the platform coordinates
$\boldsymbol{\tau}$	the vector of the platform twist
$\mathbf{q}$	the vector of active joint variables
$q_i$	the active joint variable of leg $i$
$\dot{\mathbf{q}}$	the vector of active joint velocities
$\boldsymbol{\Gamma}$	the vector of active joint torques/forces
$\mathbb{I}$	the inertia matrix
$\mathbf{A}, \mathbf{B}$	the serial/parallel Jacobians of a mechanism, such that $\mathbf{A}\dot{\mathbf{q}} + \mathbf{B}\boldsymbol{\tau} = 0$
$\mathbf{J}$	the global kinematic Jacobian of a mechanism, such that $\mathbf{J} = -\mathbf{B}^{-1}\mathbf{A}$
$\mathbf{J}^+$	the pseudo-inverse of matrix $\mathbf{J}$
$\mathbf{J}_{\text{inv}}$	the global inverse kinematic Jacobian of a mechanism, such that $\mathbf{J}_{\text{inv}} = -\mathbf{A}^{-1}\mathbf{B}$
$K_p, K_i, K_d$	the proportional, integral and derivative gains of a PID controller
$\mathcal{G}$	a function
$\mathcal{H}$	a function
${}^b\overrightarrow{AB}$	a vector from point $A$ to point $B$ , expressed in the robot base frame
${}^i\overrightarrow{AB}$	a vector from point $A$ to point $B$ , expressed in the frame of leg $i$
$\mathbf{u}_i$	a unit vector giving the direction of the axis of leg $i$

---

${}^b\mathbf{u}_i$	a unit vector giving the direction of the axis of leg $i$ , expressed in the robot base frame
${}^c\mathbf{u}_i$	a unit vector giving the direction of the axis of leg $i$ , expressed in the camera frame
${}^p\mathbf{u}_i$	a unit vector giving the direction of the axis of leg $i$ , expressed in the pixel (image) frame
$\mathbf{h}_i$	the Plücker coordinate unit vector defining the interpretation plane associated with the axis of leg $i$
$h_i$	the normalizing scalar associated with $\mathbf{h}_i$
$\mathbf{n}_i^L$	the Plücker coordinate unit vector defining the interpretation plane associated with the left edge of leg $i$ in the image
$n_i^L$	the normalizing scalar associated with $\mathbf{n}_i^L$
$\mathbf{n}_i^R$	the Plücker coordinate unit vector defining the interpretation plane associated with the right edge of leg $i$ in the image
$n_i^R$	the normalizing scalar associated with $\mathbf{n}_i^R$
${}^c\mathbf{p}$	coordinates of a point in the camera frame
${}^p\mathbf{p}$	coordinates of a point in the pixel (image) frame
$\mathbf{K}$	the matrix of intrinsic camera parameters, such that ${}^p\mathbf{p} = \mathbf{K}{}^c\mathbf{p}$
$\dot{\mathbf{u}}_i$	the time derivative of vector defining the direction of leg $i$
$\mathbf{M}_i$	the interaction matrix of leg $i$ , such that $\dot{\mathbf{u}}_i = \mathbf{M}_i\boldsymbol{\tau}$
$\alpha, \beta$	angles
$\lambda$	a scalar for the gain of the control law
$dof$	degrees of freedom
$fkp$	forward kinematic problem
$\mathbf{x}$	an axis
$\mathbf{y}$	an axis
$\mathbf{z}$	an axis
$\mathbf{v}$	an axis
$\mathbf{w}$	an axis

# Introduction

Compared to serial robots, parallel kinematic manipulators ([Leinonen, 1991](#)) are stiffer and can reach higher speeds and accelerations ([Merlet, 2006b](#)). Therefore, they are prudently considered as promising alternatives for many modern material processing operations, especially in automotive and aerospace industry, in which high accuracy positioning and high-speed motions of a work tool are required. 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.

Because of this, it is important to perform a rigorous study during design in order to produce optimal robot architectures. However, the concepts that influence the performance of the manipulator are not solely mechanical. Indeed, the control scheme used for the robot task can have a significant impact on speed, accuracy, dampening of vibrations. Consequently, the question of control should not be a matter of implementation (i.e. once the prototype is ready), but should instead be taken into account during the earliest stages of the design process.

When the specifications of the robot include extreme accuracy, it is important to use highly detailed models during usual encoder-based control, which take into account flexibility of links, deformation under high loads, and vibrations that occur at high accelerations. Also, one must take into account that even highly detailed models suffer when they are implemented, due to manufacturing and assembly errors. To make sure that the finished product corresponds to the designed models, the identification of several parameters must be performed, which in the case of mass-produced robots can be costly and time-consuming.

Such difficulties can be overcome through the use of an alternate approach: sensor-based design and control. This work exploits the advantages of using external sensors by focusing on a particular control method: observing the legs of parallel robots. Although the use of external sensors for the control of robots is not a new concept, the use of this method resulted in unexplained behaviour and unanswered questions. The main considerations were:

- It was possible to control a robot with  $n$  legs, by observing only  $m$  leg directions ( $m < n$ ). What is the physical explanation behind this? If it is possible to fully control a parallel robot by observing only a subset of its legs, how do we choose which subset to observe? Is there an optimal set of legs to observe? If yes, can we find it?
- In some cases, the robot end-effector did not converge to the desired position, despite all the observed legs converging to their desired orientations. What is the meaning behind this?
- What can we know about the existence of singularities in the mapping between the leg direction space and the Cartesian space? If singularities exist, what are their effects and can we locate and avoid them if necessary?
- What are the effects of stacking observation matrices (i.e. observing all legs versus the minimum necessary subset)? Can we be sure that this approach does not lead to local minima in the Cartesian space (for which the error in the observation space is non zero while the robot platform cannot move)?

The present work gives proper explanations to all the questions raised through the use of this particular kind of observation and helps certify the controllability of the mechanism under these circumstances. Then, by taking into consideration the control method to be used from the early stages of the design process, we can ensure that there is a connection between the designer's concepts and automatician's needs. This is accomplished through the *hidden robot concept*.

The main contributions of this work are:

- **Formalisation and generalisation of the hidden robot concept.** The hidden robot concept is a tangible visualisation of the mapping between the observation space and Cartesian space of parallel robots controlled through the use of leg direction-based visual servoing. Rigorous construction rules are developed which allow anyone to obtain the hidden robot associated with any given real robot. The significance of the concept comes from its dual nature of providing internal information about the robot being studied (i.e. its configuration) through the observation of external properties (i.e. the direction of the legs). This makes it equally useful for both the fields of mechanics and control. The concept is generalised in a double manner. First, a generalisation is set in place through the rules which guide one to obtaining the hidden robot model associated with a given real robot architecture. It is then applied to different robot families with consistent results. Second, there is a generalisation with regards to the observation being made. In the present work, cameras are used as the external sensors, which give information about the leg edges, but the concept is equally valid for any kind of external sensor observing any kind of feature of the robot.
- **Application of the hidden robot to the analysis of the controller.** The hidden robot concept is a tool which can be used very effectively to study vision-based control. First of all, it explains many phenomena which occur when a robot is being controlled using external sensors (detailed in Chapter 3). Though these can be explained mathematically,



---

the hidden robot provides a tangible physical explanation, which can be obtained through direct geometric means. Second, due to the fact that the hidden robot is in fact a parallel mechanism, well grounded tools already used in the mechanical community can be applied to it, the results of which provide clarification regarding the mapping introduced through the use of external sensors. These methods further allow the study of the controllability of the mechanism, through simple geometric study of their hidden robot counterparts. In addition, the hidden robot can be used in the analysis of the visual servoing itself, by providing data regarding the optimal set of legs to be observed, as well as providing means for optimal feature selection.

- **Sensor-based design approach taking into account the hidden robot concept.** Lastly, the hidden robot provides a tool for the analysis of the control with minimal information regarding the task (i.e. all that is known is that external sensors are to be used). It can then be incorporated in the design process of the real robot, to provide a strong connection between the design and the task for which the robot would be used. The hidden robot can give information regarding the optimisation of geometric parameters, for example, but it can go as far as optimising camera placement or validating/invalidating the sensors used and/or the features being observed. Moreover, design which does not take into account the hidden robot model can encounter problems during visual servoing, due to singularities within the mapping (which can be predicted using the hidden robot).

The advantages of using external sensors to overcome the problems encountered through the use of model-based control approaches should not be neglected. The hidden robot concept provides the tools to make use of these advantages. The results presented in this work prove that the use of the hidden robot concept is not only desirable, but *necessary* when it comes to analysing and implementing external sensor-based controllers.

The work is separated into five further chapters. In **Chapter 2**, the state of the art in parallel robots and vision-based control is presented. Different robot architectures, general modelling, and classical control methods are presented. Then, basic information on vision-based control leads into the more particular leg direction-based observation. In **Chapter 3**, the hidden robot concept is thoroughly explained, starting from the philosophy behind it, to generalisation of the concept, and ending with applications on an Adept Quattro, in the form of simulations and experiments. In **Chapter 4**, the applications of the hidden robot are extending to different families of both planar and spatial robots, and it is used in conjunction with straightforward geometrical study and application of well-known methods from the mechanical community to solve problems regarding controllability and optimal leg selection. In **Chapter 5**, the hidden robot is applied to a design process meant for optimising the geometric parameters of a Five-Bar mechanism, proving that it is necessary to be taken into account when developing for tasks controlled through external sensors. Finally, in **Chapter 6**, conclusions are drawn based on the results presented herein, and future research perspectives regarding the use of the hidden robot concept are proposed.



# State of the Art in Parallel Robots and Vision-Based Control

*This chapter presents the state of the art regarding parallel robots and vision-based control. First, there is a presentation on industrial robots, detailing certain architectures which will be treated later. Also, the general kinematics and singularity studies of these parallel manipulators are presented, which will be referenced in the upcoming chapters. Second, different classical control methods are presented, and contrasted with the vision-based control. Following this, there is a presentation on camera models used to transform the 3D scene into the 2D image, as well as different ways to parametrise the features used to accomplish this external sensor approach. Finally, the previous work on leg observation-based control of the GS platform is presented, whose unanswered questions are a leaping point for the development of the hidden robot concept.*

## 2.1 Parallel Robots

### 2.1.1 What Is a Robot?

A *robot* is a reprogrammable, multipurpose, actuated mechanism, with a degree of autonomy, moving within its environment to perform specific tasks.

A robot usually consists of a series of segments, called *links*, which at their connection points, called *joints*, perform motions relative to one another. A robot's architecture allows the positioning and orientation of its terminal part, called the *end-effector*, onto which a task-specific tool is attached. This pose is expressed using generalised coordinates, which are the coordinates of a particular point belonging to the end-effector, expressed within the frame of the *robot base*, i.e. the part of the robot which is usually fixed to the ground.

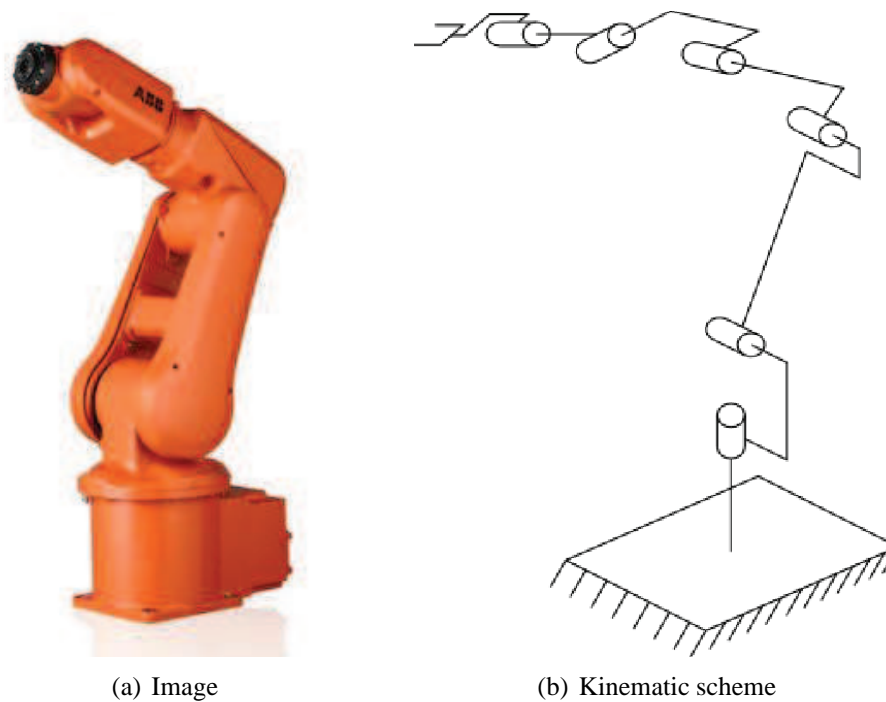


Figure 2.1 – Serial robot

The *degrees of freedom (dof)* of a robot are the minimum number of independent generalised coordinates necessary to fully express the configuration of the robot. It also describes the number of independent motions that the end-effector can perform. At maximum, these can represent three translations and three rotations, along and around the axes of a specific reference frame.

Based on their layout, we can categorise robots into *serial robots* and *parallel robots*.

### 2.1.1.1 Serial Robot Architectures

Within the industry robot manipulators are employed to substitute human workers for repetitive tasks. In most cases, serial robots are used, as these have high mobility and a large workspace, which are ideal for painting, welding, etc.

The architecture of a serial robot is composed of a series of links, each link connected to the previous and the following links. The first link is connected to the base, while the last link to the end-effector. A serial robot and its kinematic scheme are shown in Figure 2.1.

As mentioned earlier, the benefits of a serial robot are high mobility (with the ability to add additional degrees of freedom permitting accessibility to hard-to-reach points) and a large workspace. Also, the control of serial robots is usually a straight-forward issue.

However, they present some disadvantages which are crucial in the framework of this study. First of all, the small errors which are present in the joints (due to imperfections in manufacturing and assembly) accumulate due to the serial nature of the architecture. Although the individual errors are small, this accumulation results in significant loss of accuracy at the end-effector.

Secondly, each actuator has to sustain the weight of the link it is attached to, as well as all

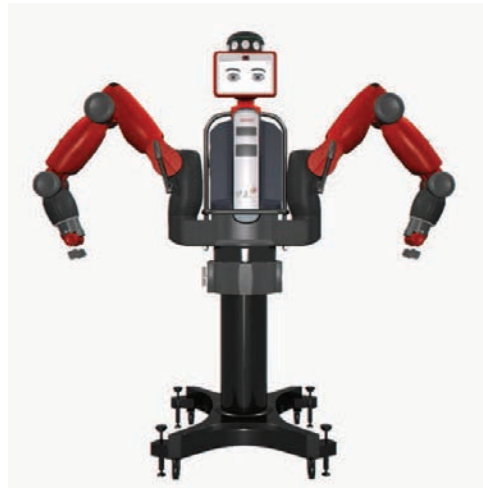


Figure 2.2 – Baxter robot

the other links that follow (and their actuators). Because of this, the payload-to-weight ratio of serial robots is quite low. It is possible to reduce this weight by moving more actuators to the base of the robot, but this requires the implementation of drive trains which further increase the errors transmitted from the actuators to the end-effector.

Thirdly, a serial architecture is not very rigid. Disturbances at the base accumulate much like the errors in the joints and have a much larger effect on the end-effector. To deal with this, CNC machines which use serial structures, for example, have massive links to confer rigidity, but because of this they also require powerful actuators, on top of losing the mobility which is an advantage of the serial architecture.

### 2.1.1.2 Tree Structure Robot Architectures

Tree-structure robots present kinematic chains which are connected to the ground, but after a certain joint split into multiple end-effectors. One such robot is the Baxter ([Guizzo and Ackerman, 2012](#)), which presents a humanoid upper body, but a simple fixed lower body (Figure 2.2). Such robots have been gaining popularity, especially in the services sector, since they can be used like humanoid robots in many situation, but are much easier to model and control, due to the lack of legs.

What is interesting to note is that tree-structure robots that manipulate objects with two or more arms at the same time can be modelled as a parallel robot.

### 2.1.1.3 Parallel Robot Architectures

Parallel robots differ from serial robots through the fact that the end-effector is attached to the base through a series of kinematic chains, forming closed loops. These kinematic chains, called legs, usually have the same structure, though they may be different in some cases. While serial architectures differ only through the choice of joints and number of links, parallel robots are more varied, having different number of legs and widely differing structures. A typical parallel robot, the Stewart platform, is shown in Figure 2.3, along with its kinematic scheme.

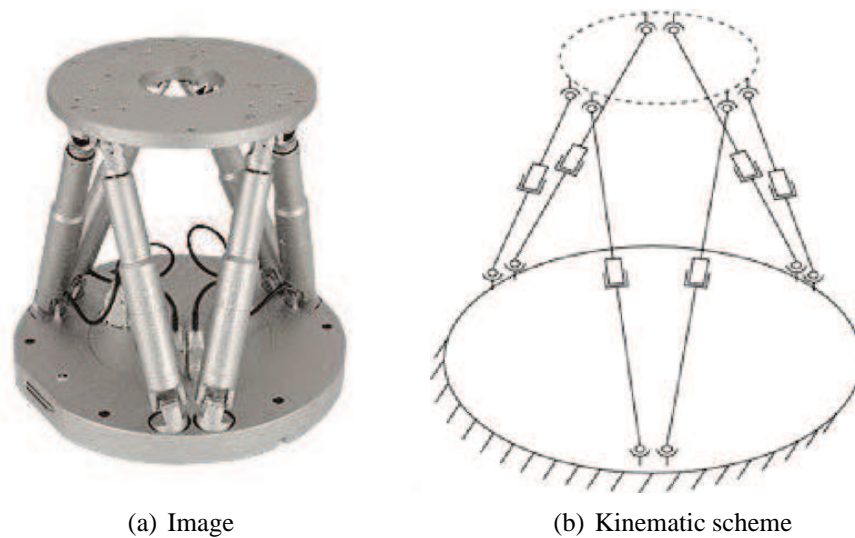


Figure 2.3 – Stewart platform (parallel robot)



Figure 2.4 – Quattro robot

It is possible to obtain full mobility (three *dof* for a planar mechanism and six *dof* for a spatial one) by having a single joint actuated on each of the legs. This allows the placement of the actuators close to the base and frees the legs from having to carry the actuators of the joints situated more towards the end-effector. Because of this, parallel structures have a greater intrinsic stiffness, present a higher payload-to-weight ratio and have better dynamic performances when compared to their serial counterparts (Tsai, 1999). As proof of this, the fastest robot used in industry to date is the Quattro (Nabat et al., 2005), which uses a parallel architecture, shown in Figure 2.4.

The main disadvantage of parallel robots is the large number of singularities within their workspace, however this topic is treated further in the report. Secondly, parallel robots have a limited workspace. Due to the fact that the end-effector is connected to the base by several kinematic chains, the workspace of the robot is a result of the intersection of each individual leg, limiting it severely when compared to serial structures.

Thirdly, parallel robots are generally harder to control than serial ones. Because each leg's contribution to the motion of the end-effector changes based on the motion of the other legs, it

is harder to decouple the motions as one might do in the case of serial robots.

However, despite these disadvantages, the better dynamic performances of parallel robots make them the desired architecture within the framework of this thesis.

## 2.1.2 Modelling Parallel Robots

Parallel robots have no dedicated modelling strategy, instead we rely on methods adapted from the modelling of serial robots. Also, as opposed to serial robots, in most cases, parallel robots exhibit a simpler (analytical) inverse model and a more complicated (usually numerical) forward model. Because of this, we will present inverse models first.

### 2.1.2.1 Geometric Modelling

**Inverse Geometric Model** Let us consider a parallel robot such as the Stewart platform in Figure 2.3. We consider that each leg  $i$  is attached to the base in point  $A_i$  and to the end-effector in point  $B_i$ . The position of the former is known in the base frame  $\{b\}$  fixed to the base of the robot in point  $O$ , while the latter is known with respect to the end-effector frame  $\{e\}$ , fixed to the mobile platform in point  $C$ . Additionally, we know the pose of the end-effector frame with respect to the base frame, i.e. the pose of the end-effector in the Cartesian space, noted as  $\mathbf{x}$ . We wish to determine the generalised coordinates of the joints, marked as  $\mathbf{q}$ , with respect to this end-effector pose.

Given the information we have, it is possible to determine the vector  $\overrightarrow{AB}$ , which is fundamental data for the inverse geometric problem.

$$\overrightarrow{AB} = \overrightarrow{AO} + \overrightarrow{OB} = \mathcal{G}_1(\mathbf{x}) \quad (2.1)$$

Also, we can determine the vector  $\overrightarrow{AB}$  along the kinematic chain of the leg, as a function of the actuated joints in the leg and the end-effector pose:

$$\overrightarrow{AB} = \mathcal{G}_2(\mathbf{x}, \mathbf{q}) \quad (2.2)$$

From here, the inverse geometric problem comes down to determining the system of equations represented by:

$$\mathcal{G}_1(\mathbf{x}) = \mathcal{G}_2(\mathbf{x}, \mathbf{q}) \quad (2.3)$$

This could prove to be a very complicated problem, however, usually the architecture of the legs of a parallel robot is quite simple and contains one or at most two actuated joints.

**Forward Geometric Model** The direct geometric model proposes to find the end-effector pose,  $\mathbf{x}$ , knowing the generalised coordinates in joint space,  $\mathbf{q}$ . This problem, although very important in the control of the mechanism, usually has no unique solution. Also, there is no algorithm to determine which of the solutions is the actual one.

By providing sensors in the passive joints, it is possible to simplify the problem, but not to a fully-determined state. Of course, depending on the architecture, there exist specialised methods to solve the problem (Merlet, 2006a). On top of these specific methods, there are also numerical algorithms, such as the iterative numerical solution detailed in (Khalil, 2012):

- using the IGM compute the current joint variables (or those corresponding to a location in the neighbourhood of the current end-effector position)  $\mathbf{q} = \text{IGM}(\mathbf{x})$
- compute the difference between the desired joint variables and the current ones  $d\mathbf{q} = \mathbf{q}^* - \mathbf{q}$
- if  $d\mathbf{q}$  is small enough, consider  $\mathbf{x}^* = \mathbf{x}$ , otherwise return to the first step
- compute the inverse Jacobian matrix  $\mathbf{J}_{\text{inv}}$
- compute the direct Jacobian matrix as a pseudo-inverse of the inverse Jacobian matrix  $\mathbf{J} = (\mathbf{J}_{\text{inv}})^+$
- compute the differential position and orientation  $\begin{bmatrix} d\mathbf{P} \\ \delta \end{bmatrix} = \mathbf{J} d\mathbf{q}$
- update the current position and orientation of the end-effector

In most cases, this algorithm is efficient and can be computed in real time. However, in our specific case of high-speed manipulators, it might prove to be too slow for computation in real time. More importantly, however, such an algorithm may sometimes not converge, or worse, converge to a different pose than the real one. This is an important limitation of the algorithm, which prompts for other methods, such as interval analysis (Merlet, 2004).

### 2.1.2.2 Kinematic Modelling

**Inverse Kinematic Model** Through the inverse kinematic model, the joint velocities as determined, knowing the velocity of the end-effector,  $\boldsymbol{\tau}$ , composed of a linear velocity vector  $\mathbf{v}$  and a rotational velocity vector  $\boldsymbol{\omega}$ .

Starting from the inverse geometric model expressed in 2.3, through derivation we obtain:

$$\mathbf{A}(\mathbf{x}, \mathbf{q})\dot{\mathbf{q}}_a + \mathbf{B}(\mathbf{x}, \mathbf{q})\dot{\mathbf{x}} + \mathbf{C}(\mathbf{x}, \mathbf{q})\dot{\mathbf{q}}_p = \mathbf{0} \quad (2.4)$$

We have to note that in most cases  $\dot{\mathbf{x}} \neq \boldsymbol{\tau}$ . Also, the above equation is clearly not unique, due to the higher number of variables. To simplify, we may choose to reduce  $\mathbf{q}$  to  $\mathbf{q}_a$ , in which case we have:

$$\mathbf{A}\dot{\mathbf{q}}_a + \mathbf{B}\dot{\mathbf{x}} = \mathbf{0} \quad (2.5)$$

In this case, both matrices are square, and provided that  $\mathbf{A}$  is invertible we get:

$$\dot{\mathbf{q}}_a = -\mathbf{A}^{-1}\mathbf{B}\dot{\mathbf{x}} = \mathbf{J}_{\text{inv}}\dot{\mathbf{x}} \quad (2.6)$$

If we consider:

$$\boldsymbol{\tau} = \mathcal{H}\dot{\mathbf{x}} \quad (2.7)$$



then:

$$\mathbf{A}\dot{\mathbf{q}}_a + \mathbf{B}\mathcal{H}^{-1}\boldsymbol{\tau} + \mathbf{C}\dot{\mathbf{q}}_p = \mathbf{0} \quad (2.8)$$

or, if we reduce  $\mathbf{q}$  to  $\mathbf{q}_a$ :

$$\mathbf{A}\dot{\mathbf{q}}_a + \mathbf{B}\mathcal{H}^{-1}\boldsymbol{\tau} = \mathbf{0} \quad (2.9)$$

Consequently:

$$\dot{\mathbf{q}}_a = -\mathbf{A}^{-1}\mathbf{B}\mathcal{H}^{-1}\boldsymbol{\tau} = \mathbf{J}_{\text{inv}k}\boldsymbol{\tau} \quad (2.10)$$

where  $\mathbf{J}_{\text{inv}k}$  is the inverse kinematic Jacobian of the robot.

A different method of calculating the inverse Jacobian is through a velocity analysis, similar to the method used in the inverse geometric analysis.

Consider the velocity of a point  $\mathbf{B}_i$  on the end-effector:

$$\mathbf{v}_{B_i} = \mathbf{v}_C + \overrightarrow{\mathbf{B}_i\mathbf{C}} \times \boldsymbol{\omega} \quad (2.11)$$

This equation may be rewritten in matrix form as a function of the end-effector velocity:

$$\mathbf{v}_{B_i} = \mathbf{J}_{B_iC}\boldsymbol{\tau} \quad (2.12)$$

However, the velocity of point  $\mathbf{B}_i$  may also be written as a function of the joint velocities (active and passive):

$$\mathbf{v}_{B_i} = \mathbf{J}_i\dot{\mathbf{q}} \quad (2.13)$$

The above two equations lead to:

$$\mathbf{J}_{B_iC}\boldsymbol{\tau} = \mathbf{J}_i\dot{\mathbf{q}} \quad (2.14)$$

By writing this equation for each of the legs, it is possible to construct the inverse Jacobian matrix which relates the end-effector velocity  $\boldsymbol{\tau}$  to the joint velocities  $\dot{\mathbf{q}}$ .

**Forward Kinematic Model** Although the computation of the inverse Jacobian is fairly simple, the calculations involved in determining the direct Jacobian are very complex.

Of course, there exist some architectures for which the computation of the direct Jacobian is not so complicated. For example, in the case of planar robots, the inverted inverse Jacobian exists ( $\mathbf{J}_{\text{inv}}^{-1}$ ) and can be calculated.

Theoretical analytic formulations of Jacobians have been proposed, but they require complicated matrix inversions.

### 2.1.2.3 Singularities

Singularities are configurations where the Jacobian matrices relating the input and the output velocities become rank deficient. These are undesirable because the number of degrees of

freedom of the system changes instantaneously (Gosselin and Angeles, 1990).

Let us write once more relationship 2.5 which we use to define the inverse kinematic model:

$$\mathbf{A}\dot{\mathbf{q}}_a + \mathbf{B}\dot{\mathbf{x}} = \mathbf{0} \quad (2.15)$$

Singularities occur when the matrices  $\mathbf{A}$  and/or  $\mathbf{B}$  become singular. Thus, we can identify three different types of singularities, which will be illustrated using the planar RRRP mechanism shown in Figure 2.5.

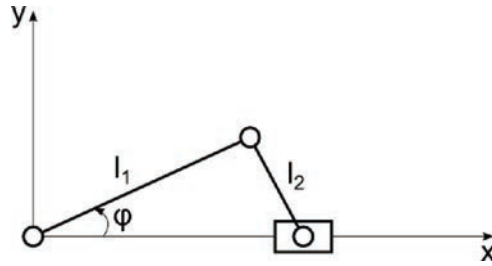


Figure 2.5 – Planar RRRP mechanism

**Type 1 Singularity** The first type of singularity occurs when:

$$\det(\mathbf{A}) = 0 \quad (2.16)$$

This corresponds to the boundary of the workspace or an internal boundary limiting the different subregions of the workspace where the number of branches is not the same.

In this case, we can find non-zero vectors  $\dot{\mathbf{q}}_a$  for which the output  $\dot{\mathbf{x}}$  is zero. Therefore, some vectors of the output cannot be produced.

In a Type 1 singularity, the output link loses one or more degrees of freedom, thus it can resist one or more forces or moments without exerting any torque or forces at the actuated joints.

In Figure 2.6 we can observe the planar RRRP in a Type 1 singularity. Notice how a force applied at the output along the  $x$  axis has no effect on the inputs.

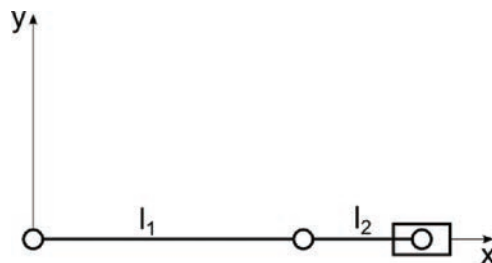


Figure 2.6 – Planar RRRP mechanism in a type 1 singularity

**Type 2 Singularity** The second type of singularity occurs when:

$$\det(\mathbf{B}) = 0 \quad (2.17)$$

As opposed to the Type 1 singularity, this configuration lies within the workspace and corresponds to a point or set of points where different branches of the direct kinematic problem meet.

In this case, some non-zero vectors  $\dot{x}$  are mapped into inputs  $\dot{q}_a$  which are zero. Therefore, we can have some output velocities which have no corresponding input velocities.

In a Type 2 singularity, the output link gains one or more degrees of freedom, thus it cannot resist one or more forces or moments even when all actuators are locked.

In Figure 2.7 we can observe the planar RRRP in a Type 2 singularity. First of all, notice that this configuration is clearly within the workspace of the mechanism. Moreover, the system cannot resist a force applied at the output along the  $x$  axis.

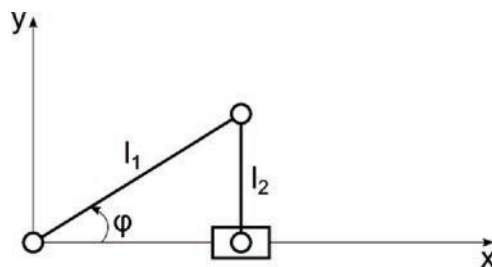


Figure 2.7 – Planar RRRP mechanism in a type 2 singularity

**Type 3 Singularity** While both Type 1 and 2 singularities can occur in general kinematic chains, Type 3 singularities require certain conditions on the geometric parameters. Because of this, they are also called *architectural singularities*.

In this case, both matrices **A** and **B** are simultaneously singular. Thus, the implicit position relation degenerates. This corresponds to configurations in which the mechanism can undergo finite motions with the actuators locked or in which finite motion of the inputs produces no motion of the outputs.

For the planar RRRP mechanism, this can occur when the input and coupler links have the same lengths. In this case, if the output  $x$  is in 0, the input can undergo arbitrary rotations without effecting the output, and the two kinds of singularities meet. This is illustrated in Figure 2.8.

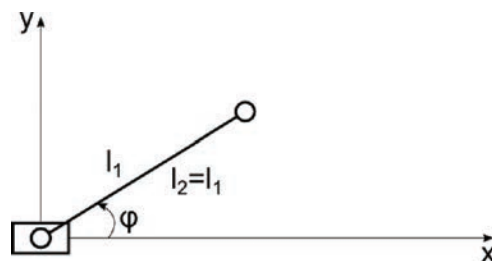


Figure 2.8 – Planar RRRP mechanism in a type 3 singularity

**Singularities Under Instantaneous Kinematics** The singularities presented above are so-called kinematic singularities. A more thorough analysis of singularities has been performed

in (Zlatanov et al., 1994) using the full range of possible instantaneous motions of a parallel manipulator with a full-cycle mobility (Hunt, 1978) of  $n$  and  $N$  total joint variables (including passive and active joints). This lead to the analysis of an  $(N + n)$ -tuple velocity vector  $\mathbf{M} = [\mathbf{T}^T \ \Omega_a^T \ \Omega_p^T]^T$ , where  $\Omega_a$ ,  $\Omega_p$  and  $\mathbf{T}$  represent the active joint velocities, the passive joint velocities and the output velocities, respectively

Under this analysis, six different types of singularities were identified:

- redundant input singularity (RI): when the motion presents a non-zero input, but a zero output;
- redundant output singularity (RO): when the motion presents a zero input, but a non-zero output (this is in fact a Type 2 singularity);
- impossible input singularity (II): when the motion admits  $n$ -dimensional vectors that cannot be applied as input;
- impossible output singularity (IO): when the motion admits  $n$ -dimensional vectors that cannot be obtained as output;
- increased instantaneous mobility (IIM): the mechanism is in an uncertainty configuration and the instantaneous mobility is greater than the full-cycle mobility;
- redundant passive motion (RPM): the mechanism is in a configuration that admits a non-zero motion with zero input and zero output.

In addition, it is possible to observe that some translational manipulators allow rotations in certain configurations (Di Gregorio and Parenti-Castelli, 1998). Since the mechanism gains a *dof* beyond its full-cycle mobility, this is a subset of IIM configurations. This particular case of singularities have been identified as constraint singularities (Zlatanov et al., 2002), which can be easily identified through the degeneration of the constraint wrench system. These is different from other IIM configurations, due to the fact that there is no choice of actuators that can prevent the platform from moving instantaneously, which makes the study of constraint singularities extremely important.

## 2.2 Classic Control Methods

### 2.2.1 PID Control

Classic PID controllers provide a linear control scheme in which the control signal is based on the difference between the desired and measured states. In general, such controllers are governed by the following control law:

$$\Gamma = K_p(\mathbf{q}^* - \mathbf{q}) + K_d(\dot{\mathbf{q}}^* - \dot{\mathbf{q}}) + K_i \int_{t_0}^t (\mathbf{q}^* - \mathbf{q}) dt \quad (2.18)$$

Proper implementation of a PID controller comes down to tuning the gains in order to obtain the desired performance. The control scheme associated with a PID controller is presented in

Figure 2.9. In addition, the integral term compensates for static disturbances (such as those caused by the effects of gravity). For mechanisms whose structure naturally compensates for this (such as planar mechanisms functioning in the horizontal plane), it is possible to remove this term and use a PD controller instead. Thus, it is possible to use a simple controller under the supposition that the dynamic behaviour exhibited by the mechanism is linear.

However, in the case of parallel mechanisms, due to the dynamic coupling between the different parts, the structure exhibits a non-linear behaviour which is significant enough that it cannot be ignored. It is possible to reduce the velocities and the workspace of the mechanism to create a region in which it does behave linearly under certain conditions, however this would restrict the already small workspace of such a robot. The solution is thus the utilisation of a controller which takes into account the non-linear dynamic behaviour of the robot, such as the computed torque control method.

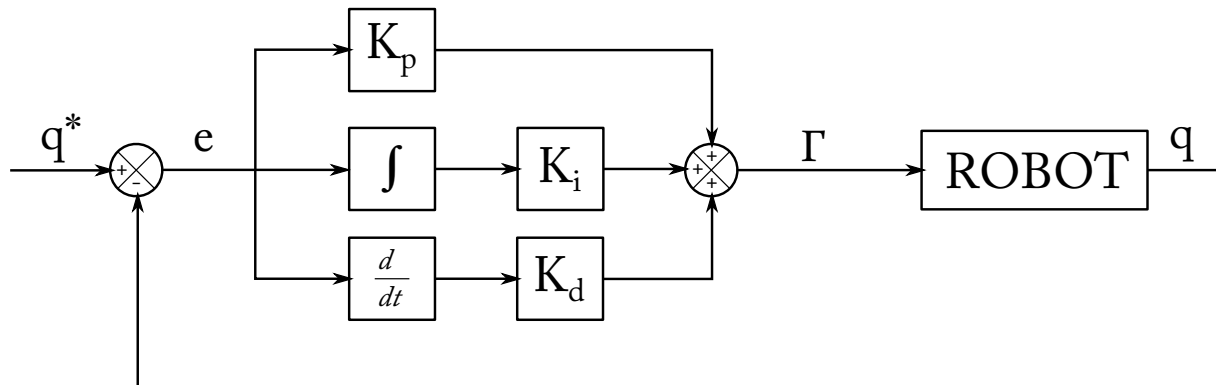


Figure 2.9 – PID control scheme

### 2.2.2 Computed Torque Control

Computed torque control is a model-based control scheme which performs an input/output linearisation using the state feedback of a non-linear system. To apply such a controller, first it is necessary to perform an input/output linearisation of the state variables, which ensures a linear behaviour of the robot with respect to the control signal. Then, a linearisation of the control signal ensures a decoupled system.

Starting from the general dynamic model of a parallel robot:

$$\Gamma = \mathbb{I}\ddot{\mathbf{q}} + \mathbf{H} \quad (2.19)$$

where  $\mathbb{I}$  is the inertia matrix (which depends on the joint variables  $\mathbf{q}$  and the platform pose  $\mathbf{x}$ ), and  $\mathbf{H}$  is a term regrouping gravitational, centrifugal and Coriolis effects (which depends on joint variables  $\mathbf{q}$  and joint velocities  $\dot{\mathbf{q}}$ ).

A straightforward linearisation can be obtained by having  $\mathbf{u} = \ddot{\mathbf{q}}$ :

$$\Gamma = \mathbb{I}\mathbf{u} + \mathbf{H} \quad (2.20)$$

This control scheme introduces a double integrator between the joint variables and the control signal. Thus, a PD controller is sufficient to control the mechanism. To ensure the convergence of the error, the following behaviour is imposed on the control signal:

$$\mathbf{u} = \ddot{\mathbf{q}}^* + K_d(\dot{\mathbf{q}}^* - \dot{\mathbf{q}}) + K_p(\mathbf{q}^* - \mathbf{q}) \quad \mathbf{u} = \ddot{\mathbf{q}}^* + K_d\dot{\mathbf{e}} + K_p\mathbf{e} \quad (2.21)$$

Finally, the obtained control law becomes:

$$\mathbf{\Gamma} = \mathbb{I}(\ddot{\mathbf{q}}^* + K_d\dot{\mathbf{e}} + K_p\mathbf{e}) \quad (2.22)$$

The scheme for the computed torque control is shown in Figure 2.10.

It must be noted that while computed torque control ensures maximal and homogeneous performances within the workspace, it is very susceptible to modelling errors. Indeed, the perturbation on the error may lead to lack of stability and accuracy. It is therefore desirable to increase model accuracy through dynamic identification (Swevers et al., 1997), use of more complex models (Kock and Schumacher, 2000), use of task-specific models (Oen and Wang, 2007), or employing robust control (Honegger et al., 2000; Vivas et al., 2003).

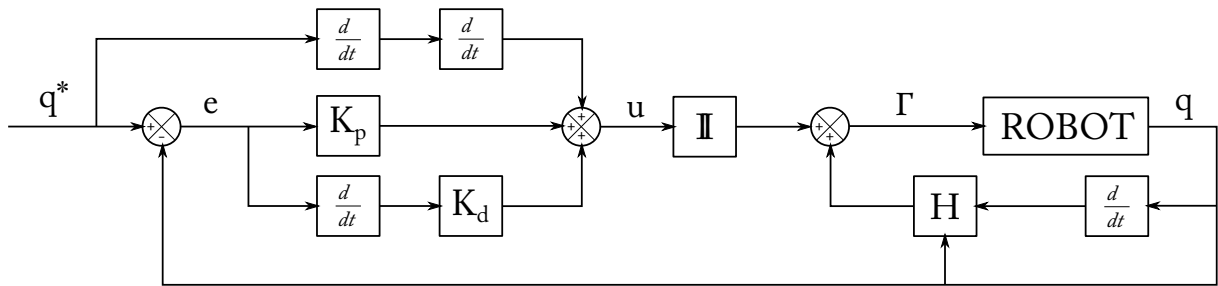


Figure 2.10 – Computed torque control scheme

### 2.2.3 Classical Control Comparison

Due to the fact that in the case of parallel robots the end-effector pose better describes the configuration of the robot, instead of the joint coordinates, it makes sense to consider the end-effector pose as the state of the robot. In this case, a Cartesian space controller instead of a joint space controller could provide better results. This, as well as the superiority of the computed torque control over a PID controller, were proven in a detailed comparison between the different control methods (Paccot et al., 2009).

As mentioned before, due to the inherent complexity of closed mechanical systems, highly non-linear behaviour must be accounted for. A single-axis PID controller cannot ensure correct performance throughout the workspace. This leads to the necessity of using the computed torque controller, which, however, must be extremely well modelled to counteract the high sensitivity of the system to the fluctuations in the error.

In the case of parallel manipulators, a Cartesian space controller is preferred over its joint space counterpart. Moreover, if the measurement of the end-effector through the use of external

sensors is possible, the modelling errors can be minimised to ensure even better performance. This is detailed in the next section.

## 2.3 Vision-Based Control

### 2.3.1 Why Use Vision?

The classical control methods presented earlier are all model based, meaning that there is a model of the robot within the controller which approximates the relationship between the motor input and the end-effector output. This is true whether position, velocity or force control is being used. The schematic of a general model-based control approach is presented in Figure 2.11.

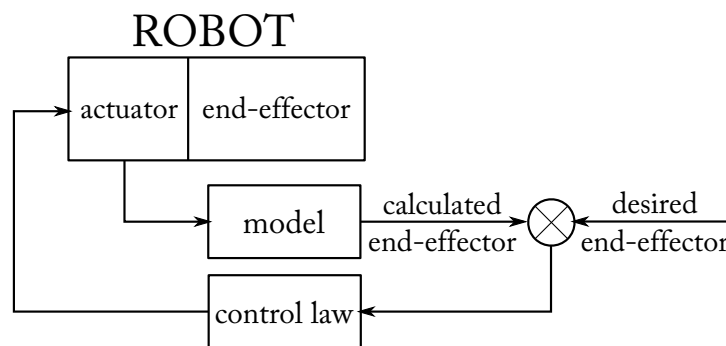


Figure 2.11 – A general model-based control scheme

It is evident then that in order to improve the performance of the control scheme, it is necessary for the model to be as accurate as possible. This sometimes involves the introduction of complex subsystems, for example to model the deformation of the links under heavy loads, which can result in high computational increase for small improvements.

Aside from deformations, assembly errors should be taken into account as well. To overcome these, proper and rigorous identification is necessary. While this can be performed on singular robots, implementing it on a large-scale to obtain extreme performances would be extremely costly and time-consuming.

As a result, it is desirable to consider a different control approach, one which is not anchored within the model. One efficient way to overcome the complexity of the model and the inconsistency errors that are associated with it is to use an external measure for the control of the robot, bypassing the model entirely. These sensor-based control approaches have proven to be more efficient than their model-based counterparts when accuracy is required in robotised industrial applications (Espiau et al., 1992). A general sensor-based control scheme is presented in Figure 2.12.

When using external sensors for control, it is important to choose an appropriate observed feature. The most common approach consists of the direct observation of the end-effector pose (Espiau et al., 1992; Horaud et al., 1998; Martinet et al., 1996). In some cases, however, it may prove difficult or unwise to observe the end-effector of the robot, e.g. in the case

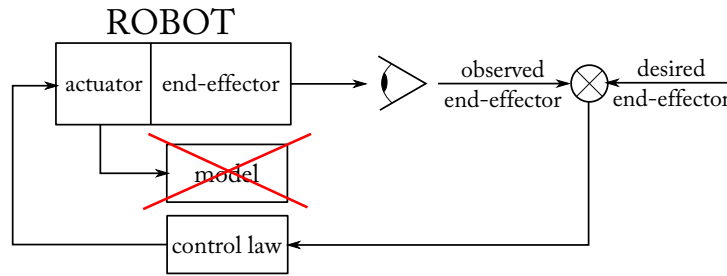


Figure 2.12 – A general sensor-based control scheme

of a milling operation. A proven alternative approach is the observation of the legs, proposed in (Andreff et al., 2005), from which the end-effector pose can be easily reconstructed.

Another important element of sensor-based control is the correct choice of sensor. For very accurate pose observation a laser tracker may be used, while accelerometers mounted on the robot legs could provide information about their accelerations. Due to the rapid evolution of the technical parameters of video cameras (in terms of frame rate and resolution), coupled with their versatility in terms of observation, makes the use of vision a prime candidate for sensor-based control.

Having opted for robot legs as the observed feature (due to the inability to observe the end-effector during typical manufacturing operations), vision was chosen as the sensor to be discussed within this work, also due to the fact that parallel robot links are usually made of rectilinear cylindrical rods that can be easily detected in the camera space. However, the methodology described herein is valid for any other type of sensor used and/or feature observed.

### 2.3.2 Basics of Vision-Based Control

The aim of all vision-based control schemes is to minimize an error  $e(t)$  (Chaumette and Hutchinson, 2008), which is typically defined by:

$$e(t) = s(\mathbf{m}(t), \mathbf{a}) - \mathbf{s}^* \quad (2.23)$$

where  $s$  represents a set of visual features, called visual primitives, which are calculated using a set of image measurements,  $\mathbf{m}$ , and additional parameters that are known about the system,  $\mathbf{a}$ .  $\mathbf{s}^*$  represents the desired values of the visual primitives.

This is a very general description of visual servoing. The different methods usually deal with the way  $s$  is defined.

The most straightforward approach is designing a velocity controller. This is based on the so-called interaction matrix  $\mathbf{L}^T$  (Chaumette, 2002) which relates the instantaneous relative motion  $T_c = {}^c\tau_c - {}^c\tau_s$  between the camera and the scene, to the time derivative of the vector  $s$  of all the visual primitives that are used through:

$$\dot{s} = \mathbf{L}_{(s)}^T T_c \quad (2.24)$$



where  ${}^c\boldsymbol{\tau}_c$  and  ${}^c\boldsymbol{\tau}_s$  are respectively the kinematic screw of the camera and the scene, both expressed in  $\mathcal{R}_c$ , i.e. the camera frame.

### 2.3.2.1 Different Approaches to Visual Servoing

**Image-Based Visual Servoing** Image-based servoing techniques use the coordinates of points in the image to define  $\mathbf{s}$ . In this case, the image measurements  $\mathbf{m}$  are the pixel coordinates of the respective points, while the additional information  $\mathbf{a}$  is represented by the intrinsic parameter matrix  $\mathbf{K}$  of the camera, which is used to transform the pixel image into 3D features.

Thus, for a 3D point with coordinates  ${}^c\mathbf{x} = [{}^c x \ {}^c y \ {}^c z]$  in the camera frame, with a corresponding projection into the image frame of coordinates  ${}^p\mathbf{x} = [{}^p x \ {}^p y]$ , we have the following relationship:

$$\begin{aligned} {}^p x &= {}^c x / {}^c z = (u - u_0) / fr \\ {}^p y &= {}^c y / {}^c z = (v - v_0) / f \end{aligned} \quad (2.25)$$

where  $\mathbf{m} = (u, v)$  represents the coordinates in pixels of the point in the image, and  $\mathbf{a} = (u_0, v_0, f, r)$  are the elements of the intrinsic camera parameters matrix  $\mathbf{K}$ , such that:

$$\mathbf{K} = \begin{bmatrix} f & 0 & u_0 \\ 0 & fr & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.26)$$

where  $f$  represents the focal length,  $r$  is the aspect ratio of the pixels, and  $u_0$  and  $v_0$  are the coordinates of the principal pixels of the image.

Taking the time derivative of the projection equations 2.25, we obtain:

$$\begin{aligned} {}^p \dot{x} &= \dot{{}^c x} / {}^c z - {}^c x \dot{{}^c z} / z^2 = ({}^c \dot{x} - {}^p x \dot{{}^c z}) / {}^c z \\ {}^p \dot{y} &= \dot{{}^c y} / {}^c z - {}^c y \dot{{}^c z} / z^2 = ({}^c \dot{y} - {}^p y \dot{{}^c z}) / {}^c z \end{aligned} \quad (2.27)$$

We can relate the velocity of the 3D point to the velocity of the camera through:

$${}^c \dot{\mathbf{x}} = -\mathbf{v}_c - \boldsymbol{\omega}_c \times {}^c \mathbf{x} \quad (2.28)$$

Inserting 2.28 into 2.27, grouping terms, and using 2.25 we obtain:

$$\begin{aligned} {}^p \dot{x} &= -v_x / {}^c z + {}^p x v_z / {}^c z + {}^p x {}^p y \omega_x - (1 + {}^p x^2) \omega_y + {}^p y \omega_z \\ {}^p \dot{y} &= -v_y / {}^c z + {}^p y v_z / {}^c z + (1 + {}^p y^2) \omega_x - {}^p x {}^p y \omega_y - {}^p x \omega_z \end{aligned} \quad (2.29)$$

which can be written as:

$$\dot{\mathbf{x}} = \mathbf{L}_x \mathbf{v}_c \quad (2.30)$$

where the interaction matrix  $\mathbf{L}_x$  is given by:

$$\mathbf{L}_x = \begin{bmatrix} -1/cz & 0 & px/cz & pxpy & -(1+px^2) & py \\ 0 & -1/cz & py/cz & 1+py^2 & -pxpy & -px \end{bmatrix} \quad (2.31)$$

**Position-Based Visual Servoing** Position-based servoing techniques use the pose of the camera with respect to some reference coordinate frame to define  $\mathbf{s}$  (Thuijot et al., 2002). In order to compute this pose from the set of measurements  $\mathbf{m}$  taken from the image, it is necessary to know additional information  $\mathbf{a}$  regarding the intrinsic parameters of the camera and the 3D model of the observed object.

It is then typical to define  $\mathbf{s}$  in terms of the parameters used to represent the camera pose. We can then have  $\mathbf{s} = (\mathbf{t}, \theta_{\mathbf{r}})$ , where  $\mathbf{t}$  is a translation vector and  $\theta_{\mathbf{r}}$  gives the angle-axis definition of the rotation.

We can now use two different approach. First, defining  $\mathbf{t}$  relative to the object frame  $\mathcal{F}_o$ . We therefore have  $\mathbf{s} = ({}^c\mathbf{t}_o, \theta_{\mathbf{r}})$ , with the desired state being  $\mathbf{s}^* = ({}^{c*}\mathbf{t}_o, 0)$ . In this case, we have an error:

$$\mathbf{e} = ({}^c\mathbf{t}_o - {}^{c*}\mathbf{t}_o, \theta_{\mathbf{r}}) \quad (2.32)$$

The interaction matrix corresponding to 2.32 is:

$$\mathbf{L}_e = \begin{bmatrix} -\mathbf{I}_c & [{}^c\mathbf{t}_o]_{\times} \\ 0 & \mathbf{L}_{\theta_{\mathbf{r}}} \end{bmatrix} \quad (2.33)$$

where  $\mathbf{L}_{\theta_{\mathbf{r}}}$  is given by (Malis et al., 1999):

$$\mathbf{L}_{\theta_{\mathbf{r}}} = \mathbf{I}_3 - \frac{\theta}{2} [\mathbf{r}]_{\times} + \left( 1 - \frac{\text{sinc}\theta}{\text{sinc}^2\frac{\theta}{2}} \right) [\mathbf{r}]_{\times}^2 \quad (2.34)$$

where  $\text{sinc}x$  is the sinus cardinal defined such that  $x\text{sinc}x = \sin x$  and  $\text{sinc}0 = 1$ .

The second approach is to define  $\mathbf{t}$  relative to the camera frame  $\mathcal{F}_c$ . In this case, we have  $\mathbf{s} = ({}^{c*}\mathbf{t}_c, \theta_{\mathbf{r}})$ , with the desired state being  $\mathbf{s}^* = 0$ , with the error being  $\mathbf{e} = \mathbf{s}$ .

The interaction matrix in this case is:

$$\mathbf{L}_e = \begin{bmatrix} \mathbf{R} & 0 \\ 0 & \mathbf{L}_{\theta_{\mathbf{r}}} \end{bmatrix} \quad (2.35)$$

This method allows for a simple control scheme, due to the decoupling between the translational and rotational motions.

**Hybrid Visual Servoing** Considering the position-based visual servoing method described earlier, we are able to obtain a decoupled control law, such that:

$$\begin{aligned} \mathbf{v}_c &= -\lambda \mathbf{R}^{Tc*} \mathbf{t}_c \\ \boldsymbol{\omega}_c &= -\lambda \theta_{\mathbf{r}} \end{aligned} \quad (2.36)$$

Then, we can combine this with an image-based visual servoing technique, by using a feature vector  $\dot{\mathbf{s}}_t$  and an error  $\mathbf{e}_t$  which are responsible for controlling the translational *dofs*, we can partition the interaction matrix, such that:

$$\begin{aligned}\dot{\mathbf{s}}_t &= \mathbf{L}_{vst} \mathbf{v}_c \\ &= \begin{bmatrix} \mathbf{L}_v & \mathbf{L}_\omega \end{bmatrix} \begin{bmatrix} \mathbf{v}_c \\ \boldsymbol{\omega}_c \end{bmatrix} \\ &= \mathbf{L}_v \mathbf{v}_c + \mathbf{L}_\omega \boldsymbol{\omega}_c\end{aligned}\quad (2.37)$$

Now, setting  $\dot{\mathbf{e}}_t = -\lambda \mathbf{e}_t$ , we can obtain a control input:

$$\begin{aligned}-\lambda \mathbf{e}_t &= \dot{\mathbf{e}}_t = \dot{\mathbf{s}}_t = \mathbf{L}_v \mathbf{v}_c + \mathbf{L}_\omega \boldsymbol{\omega}_c \\ \Rightarrow \mathbf{v}_c &= \mathbf{L}_v^+ (\lambda \mathbf{e}_t + \mathbf{L}_\omega \boldsymbol{\omega}_c)\end{aligned}\quad (2.38)$$

This method is known as the 2.5D visual servoing, first presented in (Malis et al., 1999). It shows that there can be useful applications in combining image-based and position-based visual servoing techniques.

**Eye-in-Hand and Eye-to-Hand Systems** Another important distinction that can be made when using visual servoing is the position of the camera. If the camera is mounted on the end-effector, we are dealing with a so-called *eye-in-hand* system, where the relationship between  $\dot{\mathbf{s}}$  and  $\dot{\mathbf{q}}$  is given by (Dombre and Khalil, 2010):

$$\dot{\mathbf{s}} = \mathbf{J}_s \dot{\mathbf{q}} = \mathbf{L}_s {}^c \mathbf{V}_n {}^n \mathbf{J}_n(\mathbf{q}) \dot{\mathbf{q}} \quad (2.39)$$

where  ${}^n \mathbf{J}_n(\mathbf{q})$  is the robot's Jacobian expressed in the reference frame  $\mathcal{F}_n$  of the end-effector, and  ${}^c \mathbf{V}_n$  is the kinematic torsor's transformation matrix from the camera frame  $\mathcal{F}_c$  to the end-effector frame  $\mathcal{F}_n$ .

More generally, if a mounted camera is observing a moving object, we have the following relationship:

$$\dot{\mathbf{s}} = \mathbf{L}_s {}^c \mathbf{V}_n {}^n \mathbf{J}_n(\mathbf{q}) \dot{\mathbf{q}} + \frac{\partial \mathbf{s}}{\partial t} \quad (2.40)$$

where  $\frac{\partial \mathbf{s}}{\partial t}$  represents the variation of  $\mathbf{s}$  caused by the target's own movement.

When employing a scene camera, i.e. a camera which is not mounted on the end-effector, we are dealing with an *eye-to-hand* system. In this case, we have:

$$\dot{\mathbf{s}} = -\mathbf{L}_s {}^c \mathbf{V}_n {}^n \mathbf{J}_n(\mathbf{q}) \dot{\mathbf{q}} + \frac{\partial \mathbf{s}}{\partial t} \quad (2.41)$$

where  $\frac{\partial \mathbf{s}}{\partial t}$  represents the variation of  $\mathbf{s}$  due to a possible movement of the scene camera.

Due to the fact that in this case the matrix  ${}^c \mathbf{V}_n$  is variable and has to be estimated at each iteration, a mounted camera is preferred in most applications, since in an eye-in-hand system the transformation from the camera frame to the end-effector frame can be estimated easily and

accurately using hand-eye calibration (Horaud and Dornaika, 1995; Tsai and Lenz, 1989).

### 2.3.2.2 Using Plücker Coordinates to Parametrise Lines

As mentioned in Section 2.3.1, sometimes it can be difficult to observe the end-effector through vision (e.g. in the case of milling, welding, etc.). Because of this, we have chosen to instead observe the legs of the robot and reconstruct the end-effector pose from there. In what follows is a description of how to obtain the leg direction from the image and how visual servoing using leg-observation is accomplished.

**Line Modelling** A line  $\mathcal{L}$  in space, expressed in the camera frame, is defined by its Binormalized Plücker coordinates (Andreff et al., 2002):

$$\mathcal{L} \equiv ({}^c\mathbf{u}, {}^c\mathbf{n}, {}^cn) \quad (2.42)$$

where  ${}^c\mathbf{u}$  is the unit vector giving the spatial orientation of the line,  ${}^c\mathbf{n}$  is the unit vector defining the so-called interpretation plane of line  $\mathcal{L}$  and  ${}^cn$  is a non-negative scalar. The latter are defined by  ${}^cn{}^c\mathbf{n} = {}^c\mathbf{P} \times {}^c\mathbf{u}$  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 (Plücker, 1865; Merlet, 2006b) are the couple  $({}^c\mathbf{u}, {}^cn{}^c\mathbf{n})$ .

The projection of such a line in the image plane, expressed in the camera frame, has for characteristic equation (Andreff et al., 2002):

$${}^c\mathbf{n}^T {}^c\mathbf{p} = 0 \quad (2.43)$$

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  ${}^p\mathbf{n}$  from:

$${}^p\mathbf{n}^T {}^p\mathbf{p} = 0 \quad (2.44)$$

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

$${}^p\mathbf{n}^T \mathbf{K}{}^c\mathbf{p} = 0 \quad (2.45)$$

By identification of (2.43) and (2.44), one obtains

$${}^p\mathbf{n} = \frac{\mathbf{K}^{-T}{}^c\mathbf{n}}{\|\mathbf{K}^{-T}{}^c\mathbf{n}\|}, \quad {}^c\mathbf{n} = \frac{\mathbf{K}^T{}^p\mathbf{n}}{\|\mathbf{K}^T{}^p\mathbf{n}\|} \quad (2.46)$$

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 centre (i.e. the intersection of the image diagonals) and such that the pixel coordinates vary approximately between  $-1$  and  $+1$ ,

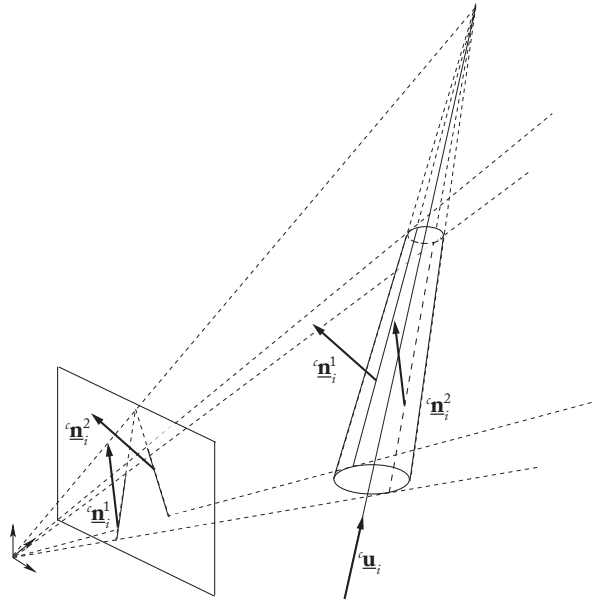


Figure 2.13 – Projection of a cylinder in the image

according to the choice of the normalizing factor, which can be the image horizontal dimension in pixels, or its vertical dimension, or its diagonal.

**Cylindrical Leg Observation** The legs of parallel robots have usually cylindrical cross-sections (Merlet, 2006b). The edges of the  $i$ -th cylindrical leg are given, in the camera frame, by (Andreff et al., 2007) (Fig 2.13):

$${}^c \underline{\mathbf{n}}_i^1 = -\cos \theta_i {}^c \underline{\mathbf{h}}_i - \sin \theta_i {}^c \underline{\mathbf{u}}_i \times {}^c \underline{\mathbf{h}}_i \quad (2.47)$$

$${}^c \underline{\mathbf{n}}_i^2 = +\cos \theta_i {}^c \underline{\mathbf{h}}_i - \sin \theta_i {}^c \underline{\mathbf{u}}_i \times {}^c \underline{\mathbf{h}}_i \quad (2.48)$$

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

It was also shown in (Andreff et al., 2007) that the leg orientation, expressed in the camera frame, is given by

$${}^c \underline{\mathbf{u}}_i = \frac{{}^c \underline{\mathbf{n}}_i^1 \times {}^c \underline{\mathbf{n}}_i^2}{\|{}^c \underline{\mathbf{n}}_i^1 \times {}^c \underline{\mathbf{n}}_i^2\|} \quad (2.49)$$

Let us remark now that each cylinder edge is a line in space, with Binormalized Plücker coordinates expressed in the camera frame  $({}^c \underline{\mathbf{u}}_i, {}^c \underline{\mathbf{n}}_i^j, {}^c n_i^j)$  (Fig 2.13). 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, a cylinder edge is entirely defined by the following constraints, expressed here in the camera frame, although valid in any frame:

$${}^c \underline{\mathbf{n}}_i^{jT} {}^c \mathbf{A}_i = -R_i \quad (2.50)$$

$${}^c \underline{\mathbf{n}}_i^{jT} {}^c \underline{\mathbf{n}}_i^j = 1 \quad (2.51)$$

$${}^c \underline{\mathbf{u}}_i^T {}^c \underline{\mathbf{u}}_i^j = 0 \quad (2.52)$$

### 2.3.2.3 Visual Servoing Using Leg Direction

For the visual servoing of a robot, one achieves exponential decay of an error  $\mathbf{e}(\mathbf{s}, \mathbf{s}^*)$  between the current primitive vector  $\mathbf{s}$  and the desired one  $\mathbf{s}^*$  using a proportional linearising and decoupling control scheme of the form:

$$T_c = \lambda \hat{\mathbf{L}}_{(s)}^{T+} \mathbf{e}(\mathbf{s}, \mathbf{s}^*) \quad (2.53)$$

where  $T_c$  is used as a pseudo-control variable and the superscript “+” corresponds to the matrix pseudo-inverse.

The proposed control approach was to servo the leg directions  ${}^c \underline{\mathbf{u}}_i$  (Andreff et al., 2005). In the case where we want to directly control the leg directions  ${}^c \underline{\mathbf{u}}_i$ , and if the camera is fixed, (2.24) becomes:

$${}^c \dot{\underline{\mathbf{u}}}_i = \mathbf{M}_i^T {}^c \boldsymbol{\tau}_c \quad (2.54)$$

where  $\mathbf{M}_i^T$  is the interaction matrix for the leg  $i$ .

The visual primitives being unit vectors, it is theoretically more elegant to use the geodesic error rather than the standard vector difference. Consequently, the error grounding the proposed control law will be:

$$\mathbf{e}_i = {}^c \underline{\mathbf{u}}_i \times {}^c \underline{\mathbf{u}}_i^* \quad (2.55)$$

where  ${}^c \underline{\mathbf{u}}_i^*$  is the desired value of  ${}^c \underline{\mathbf{u}}_i$ .

It can be proven that, for spatial parallel robots, matrices  $\mathbf{M}_i$  are in general of rank 2 (Andreff et al., 2005) (for planar parallel robots, they are of rank 1). As a result, for spatial robots with more than 2 *dof*, the observation of several independent legs is necessary to control the end-effector pose. An interaction matrix  $\mathbf{M}^T$  can then be obtained by stacking  $k$  matrices  $\mathbf{M}_i^T$  of  $k$  legs.

Finally, a control is chosen such that  $\mathbf{e}$ , the vector stacking the errors  $\mathbf{e}_i$  associated to of  $k$  legs ( $k = 3 \dots 6$ ), decreases exponentially, i.e. such that

$$\dot{\mathbf{e}} = -\lambda \mathbf{e} \quad (2.56)$$

Then, introducing  $\mathbf{N}_i^T = -[{}^c \underline{\mathbf{u}}_i^*]_{\times} \mathbf{M}_i^T$ , where  $[{}^c \underline{\mathbf{u}}_i^*]_{\times}$  is the cross product matrix associated with the vector  ${}^c \underline{\mathbf{u}}_i^*$ , the combination of (2.55), (2.54) and (2.56) gives

$${}^c \boldsymbol{\tau}_c = -\lambda \mathbf{N}^{T+} \mathbf{e} \quad (2.57)$$

where  $\mathbf{N}^T$  can be obtained by stacking the matrices  $\mathbf{N}_i^T$  of  $k$  legs. The conditions for the rank deficiency of matrix  $\mathbf{N}^T$ , as well as the conditions that lead to local minima (Chaumette, 1998) of the Equation (2.57) are discussed in Section 3.2.3.

This expression can be transformed into the control joint velocities:

$$\dot{\mathbf{q}} = -\lambda^c \mathbf{J}_{\text{inv}} \mathbf{N}^{T+} \mathbf{e} \quad (2.58)$$

where  ${}^c\mathbf{J}_{\text{inv}}$  is the inverse Jacobian matrix of the robot relating the end-effector twist to the actuator velocities, i.e.  ${}^c\mathbf{J}_{\text{inv}} {}^c\boldsymbol{\tau}_c = \dot{\mathbf{q}}$ .

### 2.3.3 Vision-Based Control of the GS Platform

Consider the GS platform in Figure 2.14. It has six *UPS* legs of varying length  $q_i, i \in 1 \dots 6$ , attached to the base by *U* joints located in points  $\mathbf{B}_i$  and to the moving platform (end-effector) by *S* joints located in points  $\mathbf{A}_i$ . The inverse kinematic model is given by:

$$\forall i \in 1 \dots 6, q_i^2 = \overrightarrow{\mathbf{A}_i \mathbf{B}_i}^T \overrightarrow{\mathbf{A}_i \mathbf{B}_i} \quad (2.59)$$

expressing that  $q_i$  is the length of vector  $\overrightarrow{\mathbf{A}_i \mathbf{B}_i}$ , i.e.

$$\overrightarrow{\mathbf{A}_i \mathbf{B}_i} = q_i \underline{\mathbf{u}}_i, \text{ or also } \mathbf{A}_i + q_i \underline{\mathbf{u}}_i = \mathbf{B}_i \quad (2.60)$$

where  $\underline{\mathbf{u}}_i$  is the unit vector of the line passing through points  $\mathbf{A}_i$  and  $\mathbf{B}_i$ .

From (Merlet, 2006b), the inverse Jacobian of the GS platform, relating the end-effector twist  $\boldsymbol{\tau}_e$  to the joint velocities is given by

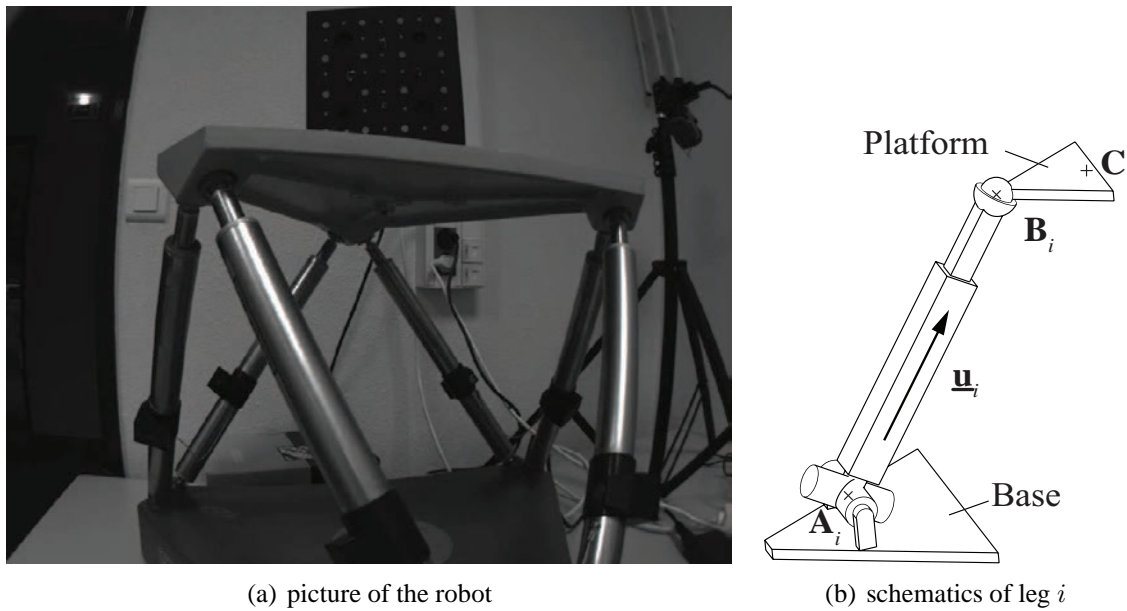
$$\dot{\mathbf{q}} = \mathbf{J}_{\text{inv}e} \boldsymbol{\tau}_e, \text{ with } \mathbf{J}_{\text{inv}e} = \begin{bmatrix} \underline{\mathbf{u}}_1^T & \left( \overrightarrow{\mathbf{C} \mathbf{B}_1} \times \underline{\mathbf{u}}_1 \right)^T \\ \vdots & \vdots \\ \underline{\mathbf{u}}_6^T & \left( \overrightarrow{\mathbf{C} \mathbf{B}_6} \times \underline{\mathbf{u}}_6 \right)^T \end{bmatrix} \quad (2.61)$$

where  $\mathbf{C}$  is the center of the end-effector reference frame  $\mathcal{R}_e$ .

The approach presented in (Andreff et al., 2005) proposed to estimate the vectors  $\underline{\mathbf{u}}_i$  using a camera. If this camera is fixed on the ground, then the reference frame associated to it is, without loss of generality, the base frame  $\mathcal{R}_b$ . As a result, the kinematics of the GS platform do not express as simply as in the end-effector embedded camera case. Indeed, expressed in the base frame, (2.61) becomes:

$${}^b\mathbf{J}_{\text{inv}e} = \begin{bmatrix} {}^b\underline{\mathbf{u}}_1^T & \left( {}^b\overrightarrow{\mathbf{C}^b \mathbf{B}_1} \times {}^b\underline{\mathbf{u}}_1 \right)^T \\ \vdots & \vdots \\ {}^b\underline{\mathbf{u}}_6^T & \left( {}^b\overrightarrow{\mathbf{C}^b \mathbf{B}_6} \times {}^b\underline{\mathbf{u}}_6 \right)^T \end{bmatrix} \quad (2.62)$$

where  ${}^b\overrightarrow{\mathbf{C}^b \mathbf{B}_i} = {}^b\mathbf{R}_e {}^e\mathbf{B}_i$ , with  ${}^b\mathbf{R}_e$  the rotation matrix between the base and end-effector frames and  ${}^e\mathbf{B}_i$  the position of point  $\mathbf{B}_i$  in the end-effector frame. Hence, it is necessary to estimate the end-effector orientation with respect to the base frame which is uncommon.



(a) picture of the robot

(b) schematics of leg  $i$ 

Figure 2.14 – A GS platform from DeltaLab.

An alternate formulation was proposed in (Andreff et al., 2005), which is well suited to visual servoing using leg observation. It consists in considering the mechanism in its dual operating mode: the end-effector is fixed and the base moves with respect to it (Figure 2.15). Thus, we are interested in the inverse Jacobian relating the base twist  ${}^b\tau_b$  expressed in the base frame to the joint velocities.

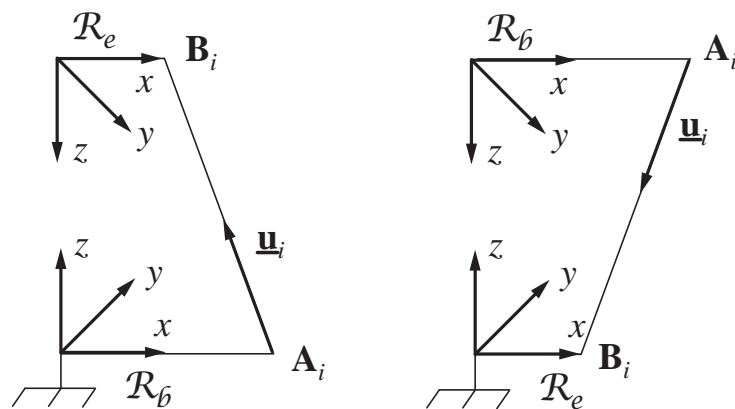


Figure 2.15 – Duality between the mobile end-effector mode and the fixed end-effector mode.

By analogy with (2.62), i.e. by permutation of the roles of  $B_i$  and  $A_i$  and of  $R_e$  and  $R_b$  (Figure 2.15), one obtains the vision-based kinematics of the GS platform expressed in the base frame (Andreff et al., 2005):

$$\dot{\mathbf{q}} = {}^b\mathbf{J}_{\text{inv}b} {}^b\tau_b, \text{ with } {}^b\mathbf{J}_{\text{inv}b} = - \begin{bmatrix} {}^b\mathbf{u}_1^T & {}^bh_1 {}^b\mathbf{h}_1^T \\ \vdots & \vdots \\ {}^b\mathbf{u}_6^T & {}^bh_6 {}^b\mathbf{h}_6^T \end{bmatrix} \quad (2.63)$$

where  ${}^bh_i {}^b\mathbf{h}_i = {}^b\mathbf{A}_i \times {}^b\mathbf{u}_i = {}^b\mathbf{B}_i \times {}^b\mathbf{u}_i$ .



**Questions Raised by Leg Direction-Based Control** We presented the application of the leg direction-based observation method to the GS platform. However, the proposed control scheme was not usual in visual servoing techniques, in the sense that in the controller, both robot kinematics and observation models linking the Cartesian space to the leg direction space are involved. As a result, some surprising results were obtained:

- the observed robot which is composed of  $n$  legs can be controlled using the observation of only  $m$  leg directions ( $m < n$ ) arbitrarily chosen among its  $n$  legs,
- in some cases, the robot does not converge to the desired end-effector pose (even if the observed leg directions did)

This last point showed that it may be possible that a global diffeomorphism between the Cartesian space and the leg direction space does not exist, but no formal proof was given. In parallel, some important questions were never answered, such as:

- How can we be sure that the stacking of the observation matrices cannot lead to local minima (for which the error in the observation space is non zero while the robot platform cannot move ([Chaumette, 1998](#))) in the Cartesian space?
- Are we sure that there is no singularity in the mapping between the leg direction space and the Cartesian space?

## 2.4 Conclusions

This chapter introduces the state of the art regarding parallel robots and vision-based control.

A general analysis of parallel robots is presented, as well as classical model-based control methods, along with their shortcomings. The more robust of these controllers are very sensitive to model-based errors. In order to eliminate these errors, it is possible to use exteroceptive sensors to directly measure the end-effector pose.

Vision-based control is introduced, starting with the typical observation of the end-effector. When it is not efficient to observe the end-effector, alternative methods should be used, which in this case is leg direction observation. It is simple to extract lines from the image and due to the cylindrical shape of parallel robot legs, obtaining their leg-directions is trivial. However, when controlling robots using their leg directions, it is possible to stumble upon unusual convergence problems, as was explained in the case of the GS platform. Moreover, other unanswered questions arise from this unconventional controller approach.

The following chapter introduces the concept of the hidden robot model, which gives answers to all these questions by providing a physical interpretation of the mapping between the leg direction space and the Cartesian space.



## The Hidden Robot Concept

*This chapter presents the hidden robot concept as a tangible visualisation of the mapping introduced by the observation of the leg directions. Preliminary observations of the hidden robot are made on the Gough-Stewart platform in light of the questions presented at the end of the previous chapter. Then, the hidden robot concept is explained, along with a general methodology for correctly constructing the hidden robot architecture corresponding to a generic real robot. Finally, advantages of using the hidden robot for the analysis of the controller are presented, which are then applied to the Adept Quattro within a case study involving numerical simulations and experimental validations.*

### 3.1 Preamble Regarding the Problems Raised by Vision-Based Control

#### 3.1.1 Description of the GS Platform's Hidden Robot Architecture

To answer the questions raised in the previous chapter, let us consider the relationship between observation and actuation. In the classical control approach, the encoders measure the motion of the actuator. In the previously described control approach (Martinet et al., 1996), the leg directions are observed. So, in a reciprocal manner, one could wonder to what kind of virtual actuators this observation corresponds.

For answering to this question, let us analyse the leg  $i$  (Figure 2.14(b)). Its unit vector  ${}^b\underline{\mathbf{u}}_i$  can obviously be parametrised by two independent coordinates that can be the angles defined by the U joint rotations. Thus,  ${}^b\underline{\mathbf{u}}_i$  is a measure of the U joint displacements. As a result, the U joint is the virtual actuator we were looking for. Observing the directions of the leg remains not to control the displacement of a UPS leg but of a virtual UPS leg with the same geometric

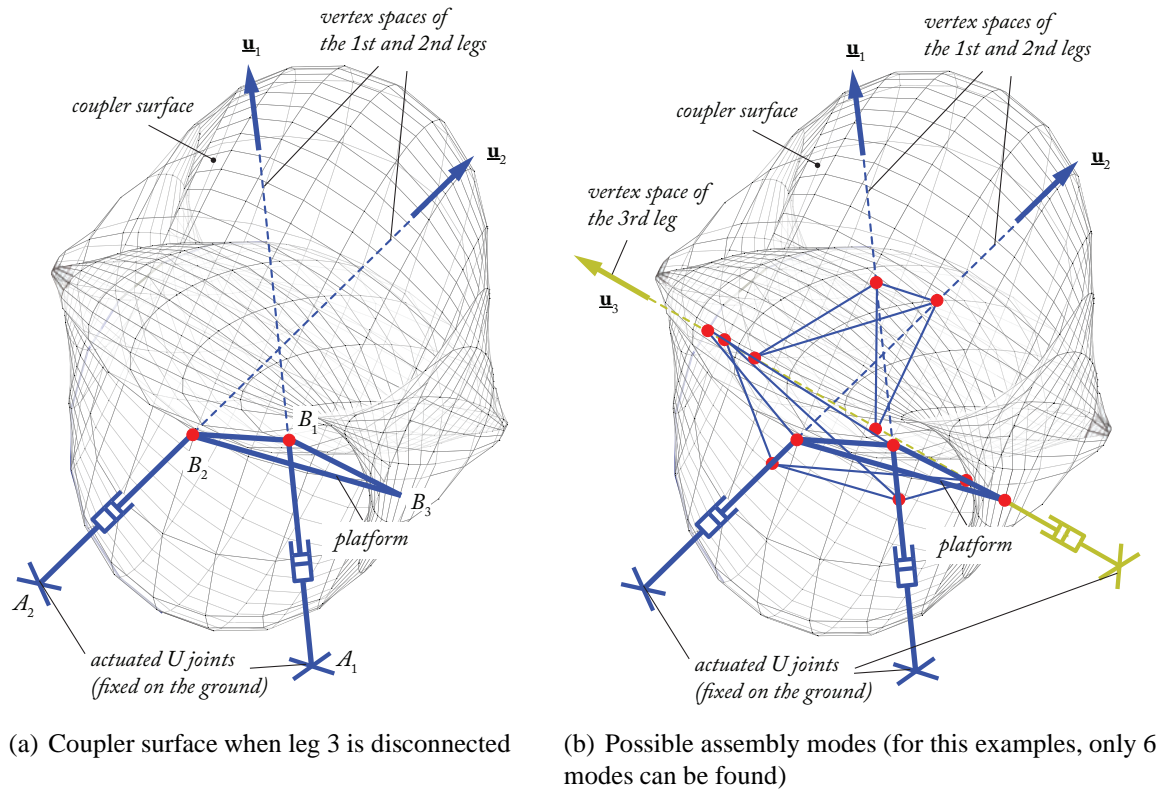


Figure 3.1 – Solutions of the  $fkp$  for a 3-UPS robot

properties as the real leg.

It is well known in the parallel robot community that a 3-UPS robot (Figure 3.2(a)) is fully-actuated. Therefore, this is the reason why it is possible to control the GS platform by observing the displacements of three of its six legs. This is equivalent to actuating a virtual 3-UPS robot with the same geometric properties as the GS platform (same attachment points, leg length, U and S joint orientations), but with assembly modes and singular configurations that differ from those of the GS platform. These should be studied in order to avoid control problems.

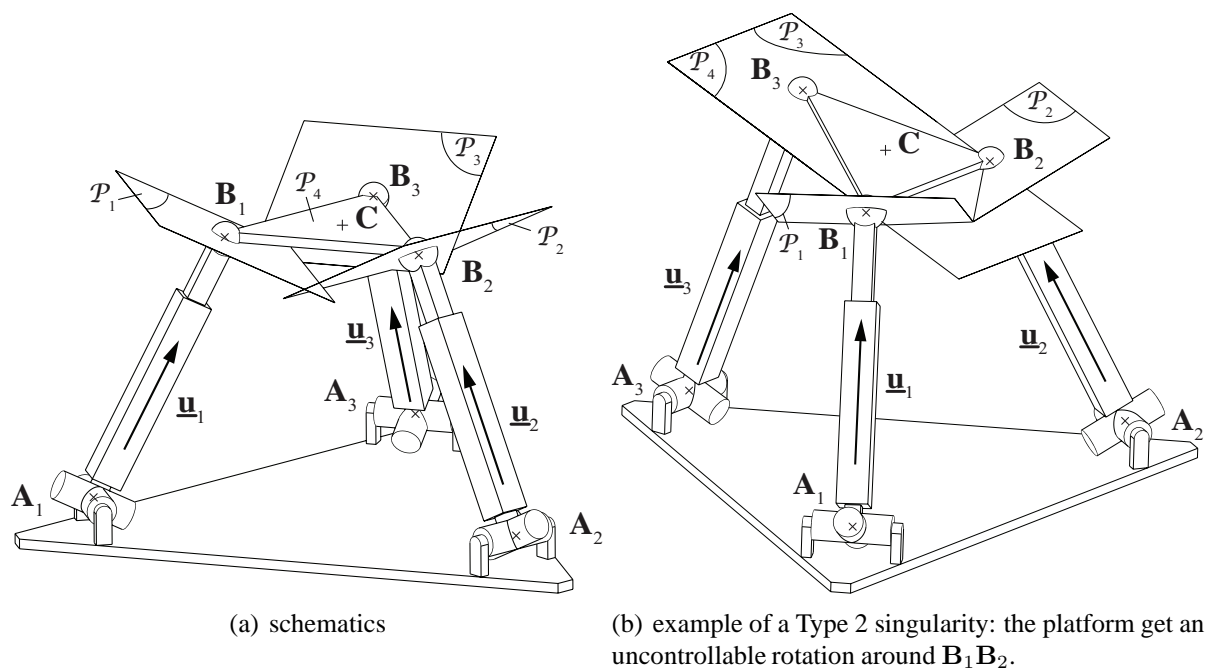
### 3.1.1.1 Forward geometric analysis

Cross multiplying the right part of (2.60) by  ${}^b\underline{u}_i$  leads to:

$${}^b\underline{u}_i \times ({}^b\underline{A}_i + q_i {}^b\underline{u}_i) = {}^b\underline{u}_i \times {}^b\underline{A}_i = {}^b\underline{u}_i \times {}^b\underline{B}_i \quad (3.1)$$

which are the equations to solve for obtaining the symbolic expressions for the 3-UPS robot forward geometric model. However, solving this problem is tedious and will not be developed here. Instead of this, let us make use of the virtual robot to employ a geometric and qualitative approach in order to better understand the forward geometric problem of this robot.

Without loss of generality, let us consider that we analyse the 3-UPS robot depicted at Figure 3.2(a). If leg 3 is disassembled at point  $B_3$ , as there are only four actuators for controlling the six robot mobilities, the platform gains two degrees-of-freedom. The gained motion is called a *spatial Cardanic motion* (Tischler et al., 1998). This motion is defined by the fact that the

Figure 3.2 – A 3-UPS robot.

points  $B_1$  and  $B_2$  are constrained to move on the lines of which directions are given by  ${}^b\underline{u}_1$  and  ${}^b\underline{u}_2$ , respectively, and the platform is free to rotate around the line  $B_1B_2$ . As demonstrated in (Tischler et al., 1998), the surface described by point  $B_3$  is an octic surface, i.e. an algebraic surface of degree eight.

As  $B_3$  also belongs to leg 3, this point is constrained to move on a line defined by the direction  ${}^b\underline{u}_3$  of the passive prismatic joint. As shown in (Tischler et al., 1998), a line and an octic surface can have up to eight real intersection points. As a result, the 3-UPS robot can have up to eight assembly modes. Let us recall here that, in the general case, the GS platform can have up to 40 assembly modes (Merlet, 2006b) that are different from those of the 3-UPS robot.

The existence of these assembly modes explains the second question presented in the previous chapter, i.e. the non systematic convergence of the end-effector to the desired pose, even if the observed leg directions do. A numerical example of this phenomenon will be presented in the Section 3.1.3.

### 3.1.1.2 Singularity analysis

The singular configurations of the 3-UPS-like robot have been deeply studied in the past (Ben-Horin and Shoham, 2006; Caro et al., 2010b). Type 1 singularities appear if one leg length  $q_i$  is equal to 0 (this is the same condition as for the GS platform). In this case, leg  $i$  can no longer produce a motion in the directions normal to  ${}^b\underline{u}_i$ . Type 2 singularities appear when the planes  $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3$  (whose normal directions are defined by the vectors  ${}^b\underline{u}_1, {}^b\underline{u}_2$  and  ${}^b\underline{u}_3$ , respectively) and the plane  $\mathcal{P}_4$  (passing through the points  $B_1, B_2$  and  $B_3$ ) intersect in one point (that can be at infinity) (Figure 3.2(b)).

Obviously, the singularity loci vary depending on the leg chosen for the GS platform control. Therefore, it is extremely important, for having the best performances of the controller, to make

an optimal selection of the three legs to observe. This is further discussed in the following section.

Finally, it should be mentioned that the singularities of the 3-UPS robot are not physical singularities, in the sense that they do not lead to uncontrollable free motions of the platform. Instead, they are representation singularities due to the mapping from the Cartesian space to the leg direction space (Ma and Angeles, 1992).

### 3.1.2 Accuracy Analysis

Several indices can be used for characterizing the neighbourhood of singularities (e.g. the condition number, the dexterity (Merlet, 2006a), the pressure angle (Arakelian et al., 2008), etc.). Here, since generally visual servoing is used for improving the robot accuracy, it is proposed to use this as an index for the characterization of singularity proximity.

From ??, and using the first order approximation of the forward geometric model (Merlet, 2006a), it is possible to write

$$\delta \mathbf{p} = \mathbf{B}^{T+} \delta \mathbf{u} \quad (3.2)$$

where  $\delta \mathbf{p} = [\delta \mathbf{x}^T, \delta \boldsymbol{\omega}^T]^T$  is the platform pose error composed of the positioning error  $\delta \mathbf{x}$  and the orientation error  $\delta \boldsymbol{\omega}$ ,  $\delta \mathbf{u}$  is the error on the observation of the leg direction, and  $\mathbf{M}^{T+}$  is the pseudo-inverse of the matrix  $\mathbf{M}^T$  that can be obtained by stacking the matrices  $\mathbf{M}_i^T$  of the three observed legs. Obviously, this matrix is the global kinematic Jacobian matrix of the equivalent 3-UPS robot and, as a result, will degenerate near the singularity configurations presented in Section 3.1.1.2. It should be mentioned here that it is decided to use a simple model for computing the robot accuracy, but any other more complicated models can be used (e.g. models that take into account flexibilities (Pashkevich et al., 2009), clearances (Binaud et al., 2010), etc.). However, this simple model is enough for our demonstration.

In the remainder of this work, the GS platform of DeltaLab is studied (Figure 2.14). This robot has the following characteristics:

$$\begin{aligned} {}^b \mathbf{A}_{2k} &= R_b \begin{pmatrix} \cos(k\frac{\pi}{3} - \alpha) \\ \sin(k\frac{\pi}{3} - \alpha) \\ 0 \end{pmatrix}, {}^b \mathbf{A}_{2k+1} = R_b \begin{pmatrix} \cos(k\frac{\pi}{3} + \alpha) \\ \sin(k\frac{\pi}{3} + \alpha) \\ 0 \end{pmatrix} \\ {}^b \mathbf{B}_{2k} &= R_e \begin{pmatrix} \cos((2k+1)\frac{\pi}{3} - \beta) \\ \sin((2k+1)\frac{\pi}{3} - \beta) \\ 0 \end{pmatrix}, {}^b \mathbf{B}_{2k+1} = R_e \begin{pmatrix} \cos((2k+1)\frac{\pi}{3} + \beta) \\ \sin((2k+1)\frac{\pi}{3} + \beta) \\ 0 \end{pmatrix} \end{aligned}$$

where  $k \in \{0, 1, 2\}$ ,  $R_b = 0.27\text{m}$ ,  $R_e = 0.195\text{m}$ ,  $\alpha = 4.25 \text{ deg.}$  and  $\beta = 5.885 \text{ deg.}$  Moreover, the leg ranges are  $[0.345 \text{ m}, 0.485 \text{ m}]$ .

For this mechanism, and for an error  $\delta \mathbf{u}_i$  defined such that the vector  ${}^b \mathbf{u}_i$  is contained in a cone of axis  ${}^b \mathbf{u}_{i0}$  and of half angle  $\phi_i$  ( ${}^b \mathbf{u}_{i0}$  is the nominal value of  ${}^b \mathbf{u}_i$  and, in what follows,  $\phi_i$  is taken equal to  $0.01 \text{ deg.}$  for each leg direction), let us compute the maximal positioning

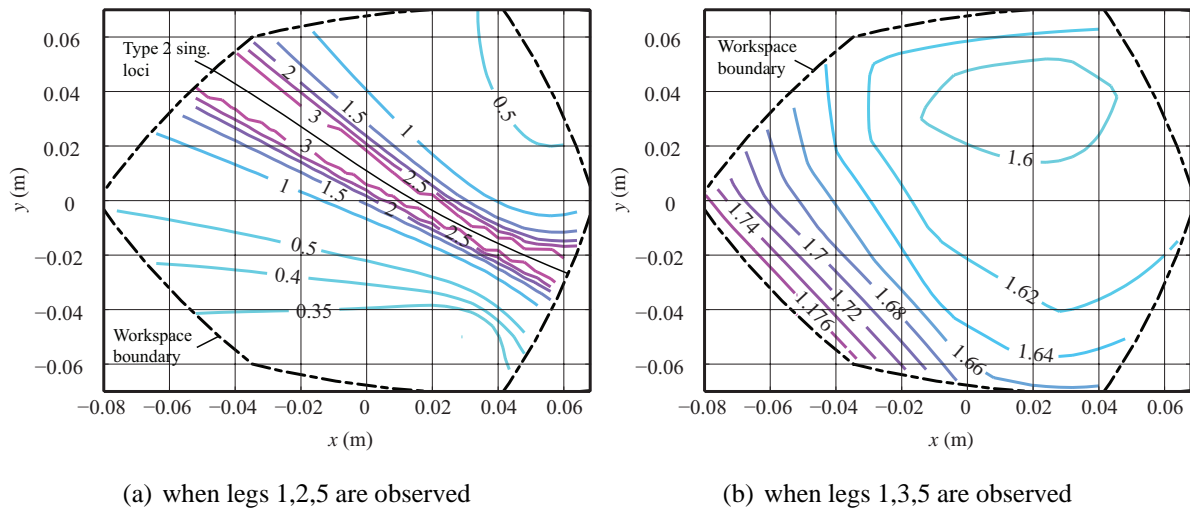


Figure 3.3 – Maximal pose error (in mm) for  $z = 0.4$  m when the platform is at zero orientation.

error when only three of its six legs are observed. Twenty different combinations are possible. However, the value of the error for only two of them (when legs 1,2,5 and 1,3,5 are observed) is plotted at Figure 3.3. In Figure 3.3(a), it is possible to note that the maximal error varies very quickly, especially near the singularity loci. In Figure 3.3(b), things are different. The variation of the accuracy is very smooth. Thus, it can be concluded that the selection of the legs to observe is crucial for the final pose accuracy.

### 3.1.3 Simulation Results

To conclude the analysis of the GS platform using the virtual 3-UPS robot, simulations are performed on an Adams mockup of the DeltaLab GS platform. This virtual mockup is connected to Matlab/Simulink via the Adams/Controls module. The controller presented in Section 2.3.2 is applied with a value of  $\lambda$  assigned to 5. The leg ranges are not considered to show the theoretical behaviour of the robot.

In the first simulations, the initial platform pose is equal to  $\{x = 0m, y = 0m, z = 0.3m, q_{r1} = 1, q_{r2} = 0, q_{r3} = 0, q_{r4} = 0\}$  and the final platform pose is set to  $\{x = -0.1m, y = 0.1m, z = 0.3m, q_{r1} = 1, q_{r2} = 0, q_{r3} = 0, q_{r4} = 0\}$  where  $q_{r1}, q_{r2}, q_{r3}, q_{r4}$  are the quaternions characterizing the platform rotations between  $\mathcal{R}_e$  and  $\mathcal{R}_b$  (Khalil and Dombre, 2002). For going from the initial point to the final ones, two sets of observed legs directions are tested:  $\{1, 2, 4\}$  and  $\{2, 3, 6\}$ . The results for the convergence of the legs directions are presented in Figure 3.4. It can be shown that when legs  $\{2, 3, 6\}$  are observed, all leg directions converge to 0. For the other case, the non observed legs do not reach their desired pose. Looking at the platform pose computed by ADAMS, the robot reaches the configuration  $\{x = -0.066m, y = 0.090m, z = 0.239m, q_{r1} = -0.931, q_{r2} = 0.290, q_{r3} = 0.101, q_{r4} = 0.197\}$  (Fig 3.5). Solving the forward geometric problem using (3.1) at the final desired robot configuration for legs  $\{1, 2, 4\}$ , it can be demonstrated that two real assembly modes exist that are  $\{x = -0.1m, y = 0.1m, z = 0.3m, q_{r1} = 1, q_{r2} = 0, q_{r3} = 0, q_{r4} = 0\}$  and  $\{x = -0.066m, y = 0.090m, z = 0.239m, q_{r1} = -0.931, q_{r2} = 0.290, q_{r3} = 0.101, q_{r4} = 0.197\}$ .



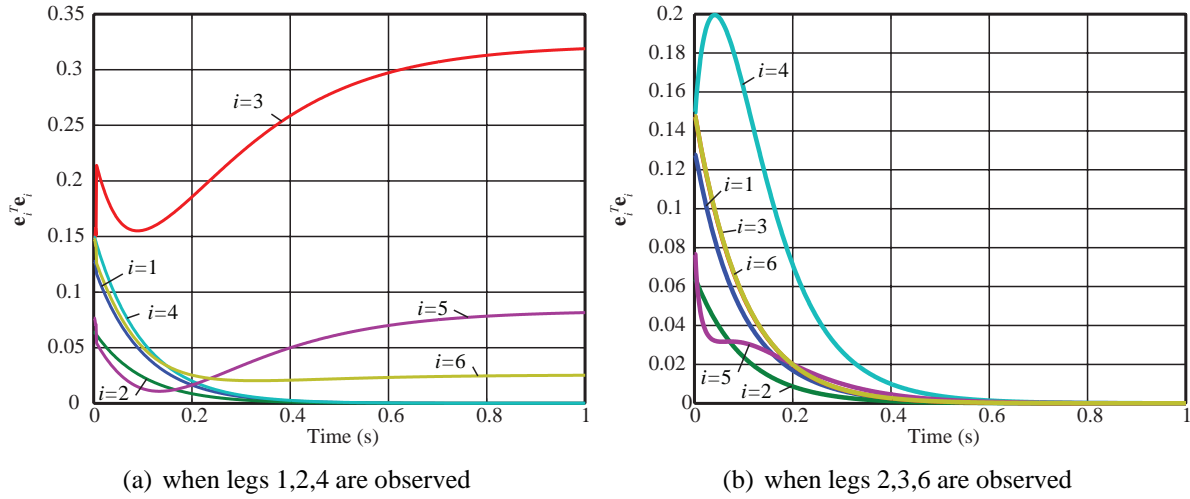


Figure 3.4 – Error on each leg  $e_i^T e_i$ .

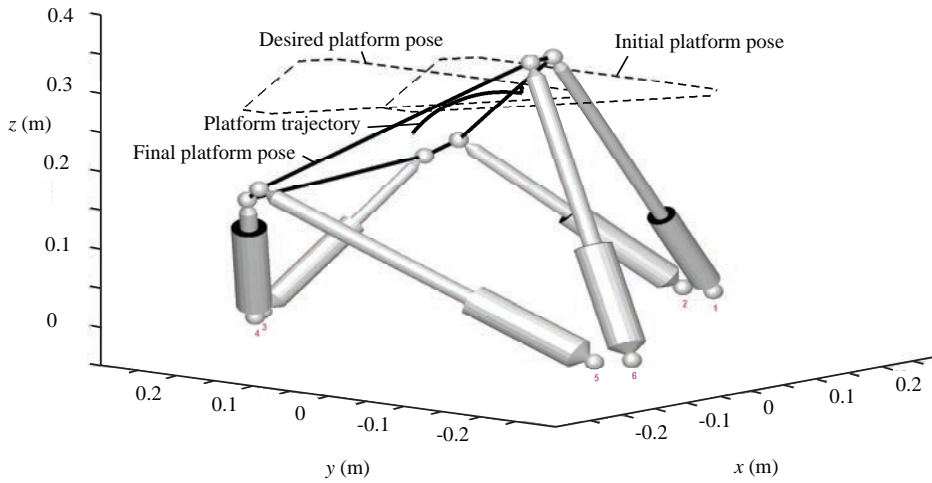


Figure 3.5 – Trajectory in space with initial, desired and final platform positions.

This validates the theory presented in Section 3.1.1.1.

In the second simulation, the final point is changed to  $\{x = 0m, y = -0.06m, z = 0.4m, q_{r1} = 1, q_{r2} = 0, q_{r3} = 0, q_{r4} = 0\}$  and a random noise of 0.01 deg is added on the simulated measure of the leg directions. To show the importance of the leg selection on the robot accuracy, it is decided to control the robot displacement using three different sets of legs: (i) legs  $\{1, 2, 5\}$ , (ii) legs  $\{1, 3, 5\}$  and (iii) all legs. The results (Figure 3.6) show that, as presented in Figure 3.15, the final platform pose accuracy is better when legs  $\{1, 2, 5\}$  are observed (around 0.3 mm) than with legs  $\{1, 3, 5\}$  (around 1.7 mm). When all legs are observed, the final pose error is much lower than when only three legs are observed. But, as mentioned in the previous section, the computational time is higher.

Using the interpretation that observing the leg directions is equivalent to controlling a virtual "hidden" robot has provided answers the questions raised at the end of the previous chapter, which were left open using conventional approaches. Moreover, this approach has proven its



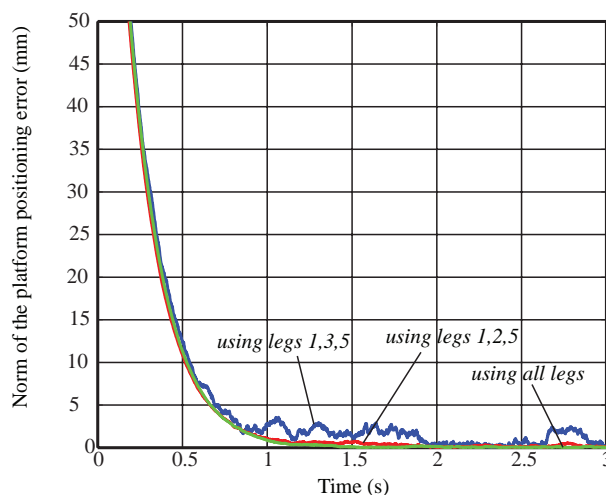


Figure 3.6 – Simulated platform pose error (in mm).

utility by providing simple geometric solutions to complicated mapping singularity problems. In what follows, this concept will be formalised, expanded upon, and applied to a different robot for validation.

## 3.2 The Hidden Robot Concept

### 3.2.1 The Philosophy Behind the Hidden Robot

The hidden robot is a concept which relates to the control of parallel robots using external sensors. The concept arises from interpreting the mapping between the Cartesian space and the observation space as a virtual robot, hidden within the controller.

The concept of hidden robot model comes from the following observation: in the classical control approach, the encoders measure the motion of the actuators and this measure is linked to the forward kinematic problem ( $fkp$ )  $\mathbf{x} = \mathcal{H}(\mathbf{q})$ , where  $\mathbf{x}$  represents the platform pose and  $\mathbf{q}$  the encoder measures; in the previously described control approach, the leg directions are observed. So, in a reciprocal manner, one could wonder to what kind of virtual actuators such observations correspond, i.e. what is the virtual robot hidden below the new  $fkp$   $\mathbf{x} = \mathcal{G}(\underline{\mathbf{u}})$ .

The utility of using the hidden robot concept is that it directly relates an external observation (the Cartesian space) to an internal configuration (the leg direction space), and then expresses this mapping through a tangible visualisation (the hidden robot itself) which can easily be analysed using well-established methods developed by the mechanical community.

### 3.2.2 How to Obtain the Hidden Robot Architecture

To determine the architecture of the hidden robot corresponding to the above-mentioned  $fkp$ , we must find the virtual actuators that are responsible for the motion of the observed feature, in our case, the leg directions.

Let us consider a general leg for a parallel robot in which the direction  $\underline{\mathbf{u}}_i$  of a segment is

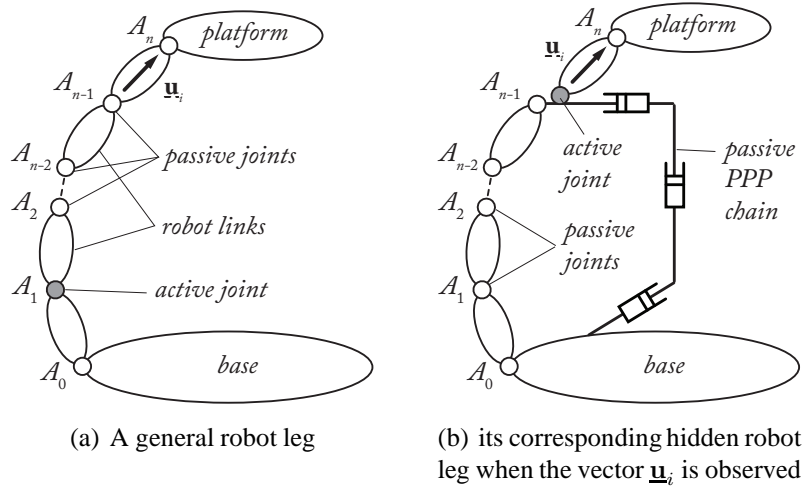


Figure 3.7 – A general robot leg and its corresponding hidden robot leg when the vector  $\underline{u}_i$  is observed

observed (Figure 3.7(a) – in this figure, the last segment is considered observed, but the following explanations can be generalized to any segment located in the leg chain). For simplification, we consider directly observing the leg direction  $\underline{u}_i$ , and not the edges in the image space. This can be done due to the fact that the edges  $\underline{n}_i^L$  and  $\underline{n}_i^R$  are only used as a measure of  $\underline{u}_i$ . We must then take into account the singularities introduced by the mapping between the leg edges and the leg direction, but this is a simple problem, since they only appear when  $\underline{n}_i^L$  and  $\underline{n}_i^R$  are collinear, i.e. the cylinders are at infinity (Andreff et al., 2007).

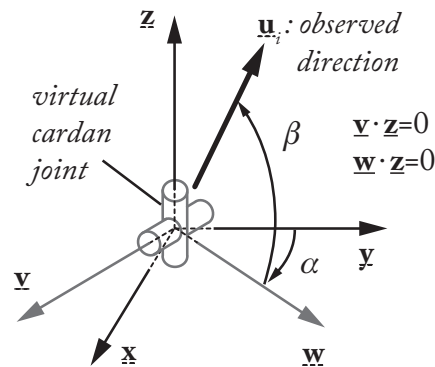


Figure 3.8 – Parametrisation of a unit vector  $\underline{u}_i$  with respect to a given frame  $x$ ,  $y$  and  $z$

In the general case, the unit vector  $\underline{u}_i$  can obviously be parametrised by two independent coordinates, that can be two angles, for example the angles  $\alpha$  and  $\beta$  of Figure 3.8 defined such that  $\cos \alpha = \underline{x} \cdot \underline{v} = \underline{y} \cdot \underline{w}$  (where  $\underline{v}$  and  $\underline{w}$  are defined such that  $\underline{z} \cdot \underline{v} = \underline{z} \cdot \underline{w} = 0$ ) and  $\cos \beta = \underline{u}_i \cdot \underline{x}$ . Thus  $\alpha$  is the angle of the first rotation of the link direction  $\underline{u}_i$  around  $\underline{z}$  and  $\beta$  is the angle of the second rotation around  $\underline{v}$ .

We must find a virtual actuator linked to the independent parameters  $\alpha$  and  $\beta$ . We need not look further than a simple U joint, which is well known to be able to orientate a link around two orthogonal axes, in this case  $\underline{z}$  and  $\underline{v}$ . Of course, other solutions may exist, but U joints with the generalised coordinates  $\alpha$  and  $\beta$  are the simplest ones. Note that when the vector  $\underline{u}_i$  is

constrained to move in a plane, such as for planar legs, the virtual actuator becomes an R joint.

If a U joint is the virtual actuator responsible for the motion of the leg direction  $\underline{\mathbf{u}}_i$ , two properties must be fulfilled:

- if the value of  $\underline{\mathbf{u}}_i$  is fixed, the U joint coordinates  $\alpha$  and  $\beta$  must also be constant, i.e. *the actuator must be blocked*,
- if the value of  $\underline{\mathbf{u}}_i$  is changing, it implies a variation in the U joint coordinates  $\alpha$  and  $\beta$ .

To ensure the aforementioned properties for  $\alpha$  and  $\beta$ , the orientation of  $\underline{\mathbf{u}}_i$  must be independent from the orientation given by the motions of the preceding links. Thus, the axes  $\mathbf{x}$ ,  $\mathbf{y}$ , and  $\mathbf{z}$  of Figure 3.8 must be the same as those of the reference frame in which  $\underline{\mathbf{u}}_i$  is expressed, be it the base frame or the camera frame. This implies that the first U joint axis must be constant with respect to the reference frame. We can distinguish two cases here: (i) when the reference frame is fixed, i.e. it is the base frame or the camera frame, which is mounted on the robot frame, and (ii) when the reference frame is mobile, i.e. it is the camera frame, when the camera is mounted on a moving link. In the first case, to ensure that the first U joint axis is constant with respect to the base frame, the U joint must be attached to a link *performing a translation with respect to the base frame*. In the second case, the U joint must perform this translation with respect to the moving link to which the camera is attached.

However, in most of the cases, the real leg architecture is not composed of U joints attached on links performing a translation with respect to the base frame. Thus, *the architecture of the hidden robot leg must be modified with respect to that of the real leg* such as depicted in Figure 3.7(b). The  $\underline{\mathbf{U}}$  joint must be mounted on a passive kinematic chain composed of at most 3 orthogonal passive P joints that ensures that the link on which it is attached performs a translation with respect to the base frame. This passive chain is also linked to the segments before the observed links so that *they do not change their kinematic properties in terms of motion*. Note that:

- it is necessary to fix the PPP chain on the preceding leg links because the information given by the vectors  $\underline{\mathbf{u}}_i$  is not enough for rebuilding the full platform position and orientation: it is also necessary to get information on the location of the anchor point  $A_{n-1}$  of the observed segment (Andreff et al., 2007). This information is kept through the use of the PPP chain fixed on the first segments;
- 3 P joints are only necessary if and only if the point  $A_{n-1}$  describes a motion in the 3D space; if not, the number of P joints can be decreased: for example, in the case of the GS platform presented in (Briot and Martinet, 2013), the U joint of the leg to control was located on the base, i.e. there was no need to add any passive P joints to keep the orientation of its first axis constant.

For example, let us have a look at the  $\underline{\mathbf{R}}\underline{\mathbf{U}}$  leg with one actuated  $\underline{\mathbf{R}}$  joint followed by a U joint of Figure 3.9(a). Using the previous approach, its virtual equivalent leg should be an  $\{\underline{\mathbf{R}}-\underline{\mathbf{P}}\}-\underline{\mathbf{U}}$  leg (Figure 3.9(b)), i.e. the  $\underline{\mathbf{U}}$  joint able to orientate the vector  $\underline{\mathbf{u}}_i$  is mounted on the top of a  $\underline{\mathbf{R}}-\underline{\mathbf{P}}\underline{\mathbf{P}}$  chain that can guarantee that:

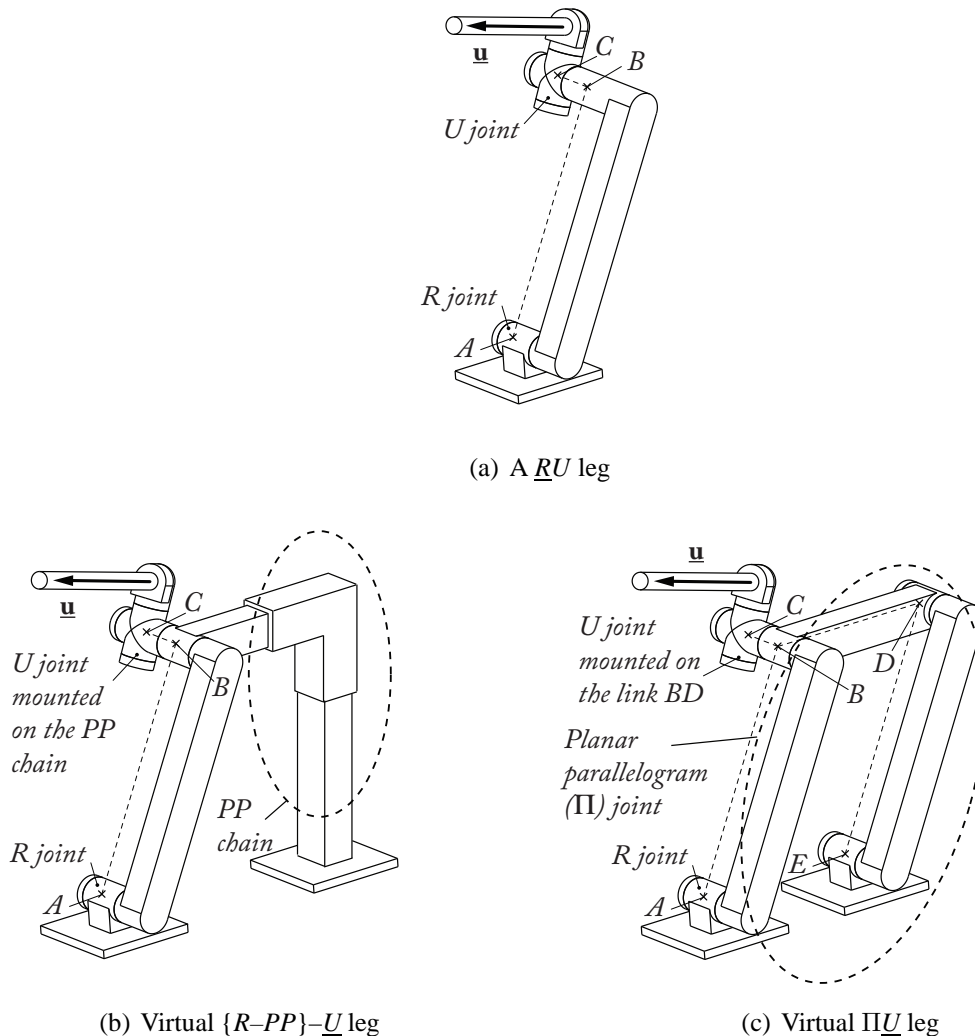


Figure 3.9 – A  $\underline{R}\underline{U}$  leg and two equivalent solutions for its hidden leg

1. the link to which the  $\underline{U}$  joint is attached performs a translation with respect to the base frame,
2. the point  $C$  (i.e. the centre of the  $\underline{U}$  joint) evolves on a circle of radius  $l_{AB}$ , like in the case of the real leg.

It should be noticed that, in several cases for robots with a lower mobility (i.e. spatial robots with a number of *dof* less than 6, or planar robots with a number of *dof* less than 3), the last joint that links the leg to the platform should be changed so that, if the number of observed legs is inferior to the number of real legs, the hidden robot keeps the same number of controlled *dof* (see Section 4.2.2).

It should also be mentioned that we have presented above the most general methodology that is possible to propose, but it is not the most elegant way to proceed. In many cases, a hidden robot leg architecture can be obtained such that fewer modifications with respect to the real leg are performed. For example, the  $R-PP$  chain of the hidden robot leg  $\{R-PP\}-\underline{U}$  (Figure 3.9(b)) could be equivalently replaced by a  $\Pi$  joint without changing the aforementioned properties of the  $\underline{U}$  virtual actuator (Figure 3.9(c)), i.e. only one additional joint is added for obtaining the

hidden robot leg (note that we consider that a  $\Pi$  joint, even if composed of several pairs, can be seen as one single joint, as in (Caro et al., 2010a)).

Because of this, there can exist multiple hidden robot leg architecture corresponding to the same robot leg. However, due to the restriction imposed by the aforementioned properties necessary to define the hidden robot leg, the kinematic behaviour of these, in terms of motion, will be the same. Thus, the different hidden leg architectures are equivalent for the purposes of our analysis, and in what follows the simplest hidden robot legs (in terms of architectural simplicity) are adopted for the studied robots.

### 3.2.3 Answers Provided by the Hidden Robot

The use of the hidden robot concept as a concrete visualisation of the mapping introduced by the use of external sensors leads to direct responses to questions previously left unanswered.

Let us consider a general 6 *dof* parallel robot composed of 6 legs, with one actuator per leg. Constructing the corresponding hidden robot using the method described above, we obtain the virtual legs each containing an actuated  $\underline{U}$  joint that has 2 degrees of actuation. For controlling the 6 *dof* of the robot only 6 degrees of actuation are needed, i.e. the observation of three virtual legs is enough to fully control the robot.

This answers the *first* question raised in Section 2.4, the possibility of controlling a robot with  $n$  legs by observing only  $m$  arbitrarily chosen legs ( $m < n$ ). By studying the rank of the interaction matrix  $\mathbf{M}$  responsible for the mapping between the Cartesian space and the leg direction space, it is possible to arrive at this same conclusion, however, the hidden robot provides an additional physical interpretation to this otherwise mathematical answer.

The hidden robot concept also makes easy to choose the optimal set of legs to observe with respect to a certain criteria, which is discussed in Section 4.3.

Due to the fact that the hidden robot may have different geometric and kinematic properties from its real counterpart, we may also observe a difference in assembly modes and singularities. If the initial and final configurations are not included in the same aspect (a singularity-free area of the workspace that is bounded by singularities (Merlet, 2006b)), the robot will not be able to converge to the desired pose, but instead will converge to a pose corresponding to another assembly mode with the same leg directions (see Figure 3.10).

This answers the *second* question raised in Section 2.4, regarding the convergence of the leg directions, despite the real robot not converging to the desired pose. The hidden robot concept shows that there does not always exist a global diffeomorphism between the Cartesian space and the leg direction space.

Moreover, the hidden robot concept can be used to avoid the convergence to a non-desired pose, as shown later in Section 4.3.

The interaction matrix  $\mathbf{M}$  involved in the controller gives the value of  ${}^c\dot{\mathbf{u}}$  as a function of  ${}^c\boldsymbol{\tau}_c$ . Thus,  $\mathbf{M}$  is the global inverse kinematic Jacobian of the hidden robot (and, consequently,  $\mathbf{M}^+$  is the hidden robot global kinematic Jacobian matrix). Except in the case of decoupled robots (Carricato and Parenti-Castelli, 2002; Kong and Gosselin, 2002; Gogu, 2004), the Jaco-

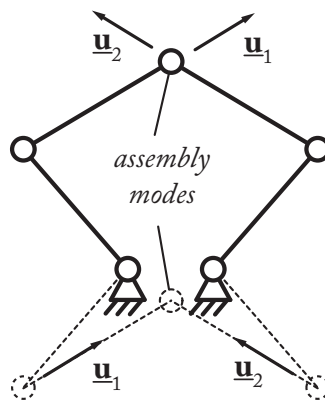


Figure 3.10 – Two configurations of a five bar mechanism for which the directions  $\underline{u}_i$  are identical (for  $i = 1, 2$ )

bian matrices of parallel robots are not free of singularities.

Considering the singularities of parallel robots presented in Section 2.1.2.3, we can conclude that:

- finding the condition for the rank-deficiency of  $\mathbf{M}$  is equivalent to finding the Type 2 singularities of the hidden robot,
- finding the condition for the rank-deficiency of  $\mathbf{M}^+$  is equivalent to finding the Type 1 singularities of the hidden robot.

This is the answer to the *third* question raised in Section 2.4, regarding the existence of singularities within the mapping between the Cartesian space and the leg direction space.

Additionally, the hidden robot concept simplifies the analysis of these singularities, by reducing the problem to the analysis of singularities of the hidden robot. This is a very powerful thing, due to the fact that for decades many tools have been developed by the mechanical design community for finding the singular configurations of robots (Bonev et al., 2003; Merlet, 2006b; Ben-Horin and Shoham, 2006; Caro et al., 2010b).

If the matrix  $\mathbf{L}^+$  of (2.57) is rank deficient, the robot could converge to local minima, i.e. points in the workspace where the control law output is nul, despite the fact that the observed legs did not converged to their desired directions. A necessary and sufficient condition for the rank deficiency of this matrix is that the  $\mathbf{M}^+$  is rank deficient, i.e. the hidden robot model encounters a Type 1 singularity.

This answers the *fourth* question raised in Section 2.4, regarding the existence of local minima within the controller. As mentioned above, many tools have been developed by the mechanical design community for finding the singular configurations of robots and solutions can be provided to ensure that the hidden robot assembly model does not meet any Type 1 singularity.

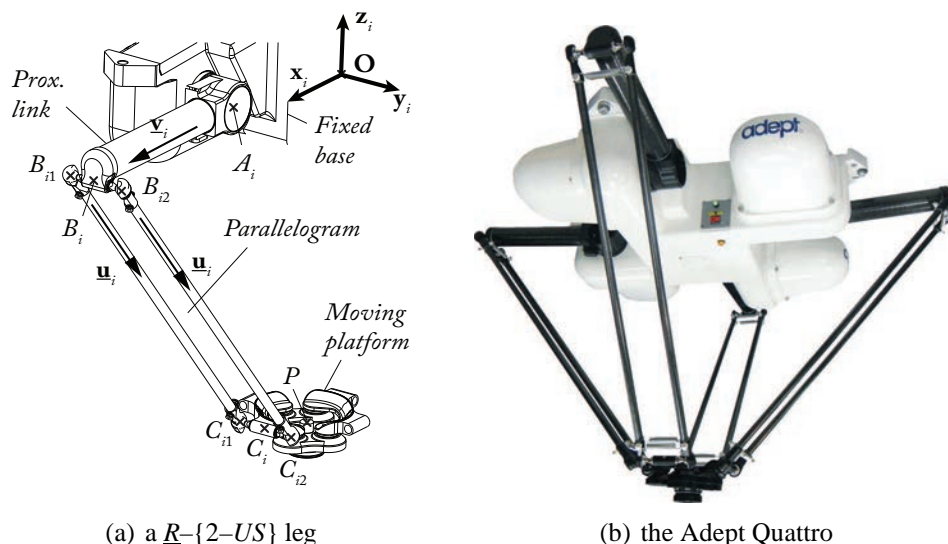


Figure 3.11 – Example of leg and of robot of the Delta-like family

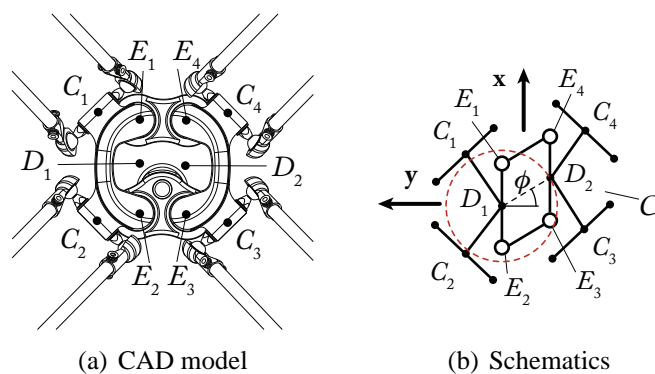


Figure 3.12 – The platform of the Quattro

### 3.3 Application of the Hidden Robot to the Adept Quattro

In this section we propose to control the Adept Quattro using leg direction-based visual servoing. As such, we are constructing the hidden robot associated with the Quattro and performing simulations in terms of accuracy, singularities, and local minima of the controller. The simulations consist of point-to-point motions of the robot platform, according to the control law presented in 2.3.2. Then, experiments are conducted, the results of which confirm the predictions of our simulations and demonstrate the existence of the hidden robot within the controller, as presented earlier in this chapter.

#### 3.3.1 Kinematics of the Quattro

##### 3.3.1.1 Usual inverse kinematics of the Quattro

The following notations are used to describe the architecture of the Quattro:

- point  $B_i$  ( $C_i$ , resp.) is at the middle of segment  $B_{i1}B_{i2}$  ( $C_{i1}C_{i2}$ , resp.) (Figure 3.11(a)),
- point  $P$ , the controlled point of the platform, is the barycenter of points  $C_i$ ; its coordinates are denoted as  $\mathbf{x}_c$  and its velocity as  $\boldsymbol{\tau}_c$ ,



- the platform orientation is parametrised by the angle  $\phi$  between the axis  $\mathbf{x}$  of the robot base frame and the vector  $\overrightarrow{D_1 D_2}$ ,
- $\mathbf{A}_i$  ( $\mathbf{B}_i$ ,  $\mathbf{C}_i$ , resp.) is the vector of coordinates of point  $A_i$  ( $B_i$ ,  $C_i$ , resp.),
- $q_i$  is the angular coordinate of the actuator  $i$ , and is defined as the angle between the axis  $\mathbf{x}_i$  (the projection of vector  $\overrightarrow{A_i B_i}$  in the horizontal plane  $Oxy$ ) and  $\overrightarrow{A_i B_i}$  around  $\mathbf{y}_i$  (Figure 3.11(a)),
- $l_1$  is the length of the proximal link, and  $l_2$  the length of one rod of the parallelogram,

The Adept Quattro has the geometric characteristics presented in Table 3.1:

Table 3.1 – Geometric characteristics of the Adept Quattro

$l_1$	0.380m
$l_2$	0.825m
${}^b \mathbf{A}_i$	$0.275 [\cos \theta_i \ \sin \theta_i \ 0]^T$ (in meters)
$\theta_i$	$\{-3\pi/4, -\pi/4, \pi/4, 3\pi/4\}$ (in radians)
${}^b \overrightarrow{D_1 C_1}$	$[-0.066 \ -0.048 \ 0]^T$ (in meters)
${}^b \overrightarrow{D_1 C_2}$	$[0.066 \ -0.048 \ 0]^T$ (in meters)
${}^b \overrightarrow{D_2 C_3}$	$[0.066 \ 0.048 \ 0]^T$ (in meters)
${}^b \overrightarrow{D_2 C_4}$	$[-0.066 \ 0.048 \ 0]^T$ (in meters)
${}^b \overrightarrow{E_1 D_1}$	$[0.057 \ 0 \ 0]^T$ (in meters)
${}^b \overrightarrow{E_4 D_2}$	$[0.057 \ 0 \ 0]^T$ (in meters)
${}^b \overrightarrow{E_2 D_1}$	$[-0.057 \ 0 \ 0]^T$ (in meters)
${}^b \overrightarrow{E_3 D_2}$	$[-0.057 \ 0 \ 0]^T$ (in meters)
${}^b \overrightarrow{PE_1}$	$0.043 [\sin(\phi + \pi/2) \ -\cos(\phi + \pi/2) \ 0]^T$ (in meters)
${}^b \overrightarrow{PE_2}$	$0.043 [\sin(\phi + \pi/2) \ \cos(\phi + \pi/2) \ 0]^T$ (in meters)

The usual inverse kinematics of the Quattro can be computed using the following loop-closure equations (Figure 3.11(b)):

$${}^i \mathbf{C}_i - {}^i \mathbf{B}_i = l_2 {}^i \underline{\mathbf{u}}_i \quad (3.3)$$

where

$${}^i \mathbf{B}_i = {}^i \mathbf{A}_i + l_1 \begin{bmatrix} \cos q_i & 0 & \sin q_i \end{bmatrix}^T = {}^i \mathbf{A}_i + l_1 {}^i \underline{\mathbf{v}}_i \quad (3.4)$$

and

$${}^i \mathbf{C}_i = {}^i \mathbf{x}_c + {}^i \overrightarrow{PC}_i \quad (3.5)$$

Squaring both sides of (3.3) and introducing (3.4) leads to

$$(x_{A_i C_i} - l_1 \cos q_i)^2 + y_{A_i C_i}^2 + (z_{A_i C_i} - l_1 \sin q_i)^2 - l_2^2 = 0 \quad (3.6)$$

where  ${}^i \mathbf{C}_i - {}^i \mathbf{A}_i = [x_{A_i C_i}, y_{A_i C_i}, z_{A_i C_i}]^T$ . (3.6) can be finally solved as a second order polynomial in  $\tan(q_i/2)$  by replacing  $\cos q_i$  with  $(1 - t_i^2)/(1 + t_i^2)$  and  $\sin q_i$  with  $2t_i/(1 + t_i^2)$ ,



where  $t_i = \tan(q_i/2)$ . Skipping all mathematical derivations, it comes that:

$$q_i = 2 \tan^{-1} \left( \frac{-\beta_i \pm \sqrt{\alpha_i^2 + \beta_i^2 - \gamma_i^2}}{\gamma_i - \alpha_i} \right) \quad (3.7)$$

where

$$\begin{aligned} \alpha_i &= -2l_1 x_{A_i C_i}, \beta_i = -2l_1 z_{A_i C_i} \\ \gamma_i &= x_{A_i C_i}^2 + y_{A_i C_i}^2 + z_{A_i C_i}^2 + l_1^2 - l_2^2 \end{aligned} \quad (3.8)$$

The first-order kinematics that relates the platform translational velocity  $\tau_c$  to the actuator velocities can be obtained through the differentiation of (3.6) with respect to time and can be expressed as:

$$\mathbf{A} \dot{\mathbf{q}} + \mathbf{B} \tau_c = \mathbf{0} \quad (3.9)$$

where  $\mathbf{A}$  is a diagonal matrix whose  $i$ -th diagonal term is

$$a_i = l_1 l_2 {}^i \underline{\mathbf{u}}_i^T {}^i \underline{\mathbf{u}}_i^\perp, {}^i \underline{\mathbf{u}}_i^\perp = \begin{bmatrix} -\sin q_i & 0 & \cos q_i \end{bmatrix}^T \quad (3.10)$$

and the  $i$ -th line of  $\mathbf{B}$  can be written as

$$\mathbf{b}_i = l_2 {}^i \underline{\mathbf{u}}_i^T \quad (3.11)$$

It should be mentioned that  $\mathbf{B}$  is a  $4 \times 3$  rectangular matrix. As a result,

$$\dot{\mathbf{q}} = -\mathbf{A}^{-1} \mathbf{B} \tau_c = \mathbf{J}_{\text{inv}} \tau_c, \text{ or also } \tau_c = \mathbf{J}_{\text{inv}}^+ \dot{\mathbf{q}} \quad (3.12)$$

where  $\mathbf{J}_{\text{inv}}^+$  is the pseudo-inverse of  $\mathbf{J}_{\text{inv}}$ .

### 3.3.1.2 Kinematics of the Quattro using leg observation

The servoing of the Adept Quattro robot using leg observation proposes to observe the parallelogram direction  $\underline{\mathbf{u}}_i$  to control the robot displacements.  $\underline{\mathbf{u}}_i$  can be obtained directly from (3.3)

$$\underline{\mathbf{u}}_i = (\mathbf{C}_i - \mathbf{B}_i) / l_2 \quad (3.13)$$

Introducing (3.4) into (3.13) and differentiating (3.13) with respect to time leads to:

$$\dot{\underline{\mathbf{u}}}_i = (\tau_c - l_1 \underline{\mathbf{v}}_i^\perp \dot{q}_i) / l_2 \quad (3.14)$$

Finally, from (3.12), it comes that:

$$\dot{\underline{\mathbf{u}}}_i = (\mathbf{I}_3 + l_1 \underline{\mathbf{v}}_i^\perp \mathbf{b}_i / a_i) / l_2 \tau_c = \mathbf{M}_i^T \tau_c \quad (3.15)$$

where  $\mathbf{I}_3$  is the  $(3 \times 3)$  identity matrix and matrix  $\mathbf{M}_i^T$  is called the interaction matrix. These

equations are valuable as long as  $a_i \neq 0$  ( $a_i = 0$  is a Type 1 singularity condition).

Note that the equation (3.15) requires the computation of the input joint variables  $q_i$  which can be estimated through the observation of the leg direction only (without any use of the encoder measurement).

It can be proven that the matrix  $\mathbf{M}_i^T$  is of rank 2. As a result, a minimum of two independent legs is necessary to control the end-effector pose. An interaction matrix  $\mathbf{M}^T$  can then be obtained by stacking the matrices  $\mathbf{M}_i^T$  of  $k$  legs ( $k = 2 \dots 4$ ). The conditions for the rank deficiency of the interaction matrix have been presented in Section 4.2.2.

The previous equations characterize the inverse kinematics of the hidden robot models of the Quattro.

### 3.3.1.3 Analysis of the hidden robot associated with the Quattro

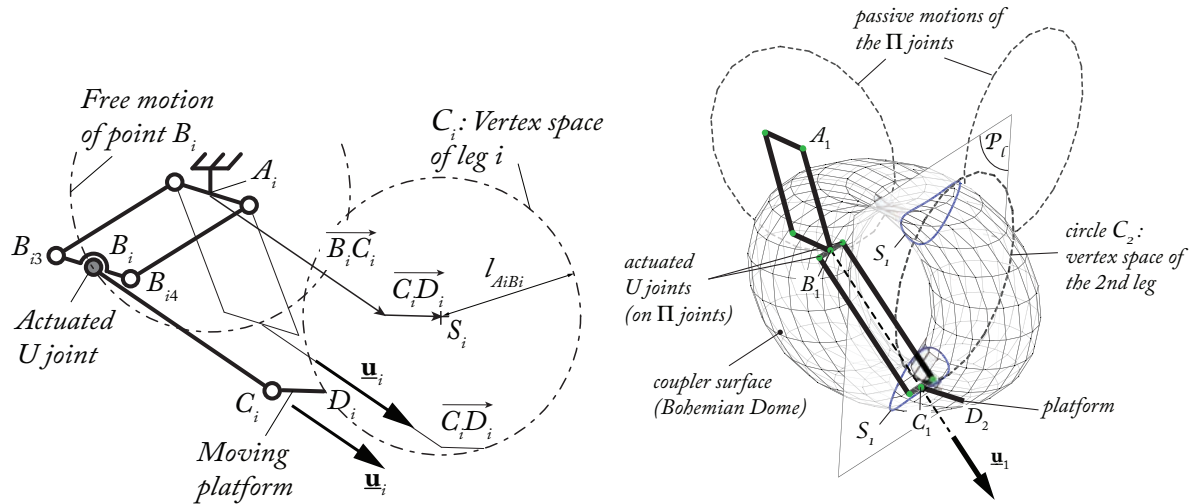
**Forward Kinematics and Assembly Modes** Looking at the vertex space of each leg when the active  $\underline{U}$  joints are fixed, the points  $C_i$  and  $D_i$  are carrying out a circle  $\mathcal{C}_i$  of radius  $l_{A_i B_i}$  centred in  $S_i$  (Figure 3.13(a)).

The Quattro with 4 *dof*, and consequently its hidden robot model, has a particularity: its platform is passively articulated (Figure 3.12) so that its orientation with respect to the horizontal plan  $Oxy$  stays constant, while it can have one degree of rotation around the  $z$  axis, i.e. point  $D_2$  can describe a circle  $\mathcal{C}_l$  located in the horizontal plane, centred in  $D_1$  and with a radius  $l_{D_1 D_2}$ . For solving the forward kinematics, it is thus necessary to virtually cut the platform at point  $D_2$  and to compute the coupler surface of point  $D_2$  when it belongs to leg 1. This coupler surface is the surface generated by  $\mathcal{C}_l$  when it performs a circular translation along  $\mathcal{C}_1$ . Such a surface is depicted in Figure 3.13(b) and is called a Bohemian Dome (Tale Masouleh et al., 2011).

A Bohemian Dome is a quartic surface, i.e. an algebraic surface of degree 4. When it intersects the vertical plane  $\mathcal{P}_l$  containing the circle  $\mathcal{C}_2$  (i.e. vertex space of the second leg), the obtained curve is a quartic curve (denoted at  $\mathcal{S}_1$  – Figure 3.13(b)). And using the Bézout theorem (Bézout, 1764), it can be proven that, when the circle corresponding to the vertex space of leg 2 intersects this quartic curve, there can exist at most 8 intersection points, i.e. 8 assembly modes. Some examples of assembly modes for the 2- $\Pi$ -{2- $\underline{U}\underline{U}$ } robot are depicted in Figs. 3.13(c) and 3.13(d).

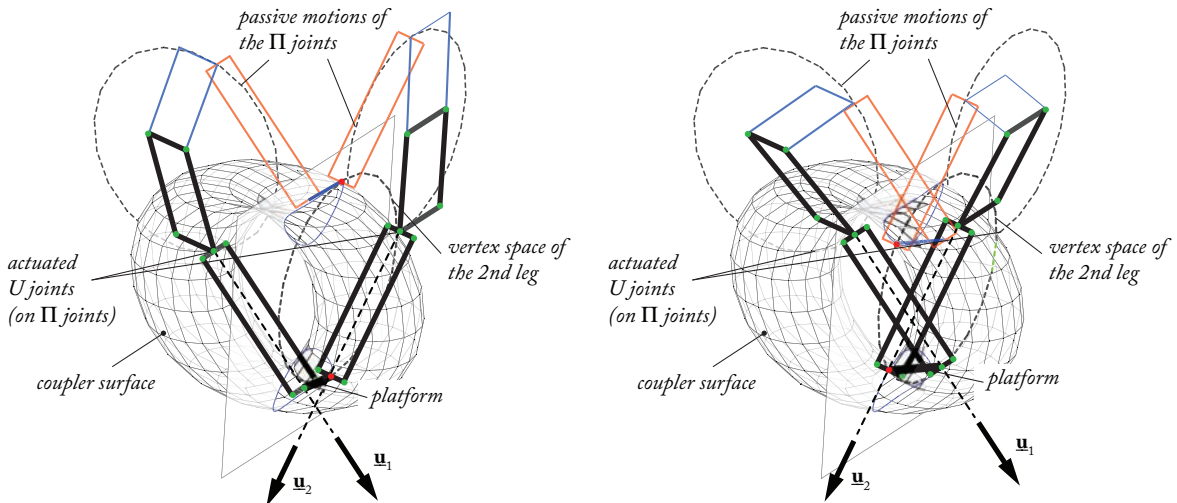
It should be noted that, when circles  $\mathcal{C}_1$  and  $\mathcal{C}_2$  are located in parallel planes,  $\mathcal{S}_1$  degenerates into 1 or 2 circles. In this case, the maximal number of assembly modes decreases to 4. It must be mentioned here that, in usual controllers when only the encoder data is used, the number of assembly modes of the Quattro is equal to 8.

**Singular Configurations** For the 2- $\Pi$ -{2- $\underline{U}\underline{U}$ } robot, Type 2 singularities appear when the planes  $\mathcal{P}_i$  and  $\mathcal{P}_j$  (whose normal vectors are equal to  $\underline{\mathbf{v}}_i^\perp$  and  $\underline{\mathbf{v}}_j^\perp$ , resp.) are parallel. In such cases, the circle  $\mathcal{C}_2$  is tangent to the Bohemian Dome at their intersection point and the robot gains one uncontrollable *dof* along this tangent (Figure 3.14).



(a) vertex space of point  $D_i$  (projection of the  $\Pi$ -{2- $\underline{U}\underline{U}$ } leg in a vertical plane)

(b) Coupler surface when leg 2 is disconnected



(c) First set of possible assembly modes

(d) Second set of possible assembly modes

Figure 3.13 – Solutions of the  $fkp$  for a 2- $\Pi$ -{2- $\underline{U}\underline{U}$ } robot (in this example, only 4 assembly modes exist)

### 3.3.1.4 Accuracy analysis of the Quattro using leg observation

For this mechanism, in the case of a leg direction based visual servoing and for an error  ${}^b\delta\mathbf{u}_i$  defined such that the vector  ${}^b\mathbf{u}_i$  is contained in a cone of axis  ${}^b\mathbf{u}_{i0}$  and of half angle  $\psi_i$  ( ${}^b\mathbf{u}_{i0}$  is the nominal value of  ${}^b\mathbf{u}_i$  and, in what follows,  $\psi_i$  is taken equal to 0.1 deg for each leg direction), let us first compute the maximal positioning and orientation error when only two of its four legs are observed. Six different combinations are possible. However, the value of the error for only two of them (when legs {2, 3} and {2, 4} are observed) is plotted at Figs. 3.15 and 3.16.

In Figs. 3.15(a) and 3.16(a), it is possible to note that the maximal error varies very quickly, especially near singularity. In Figs. 3.15(b) and 3.16(b), things are different. The variation of the accuracy is smoother for the orientation error, and the position accuracy decrease in the

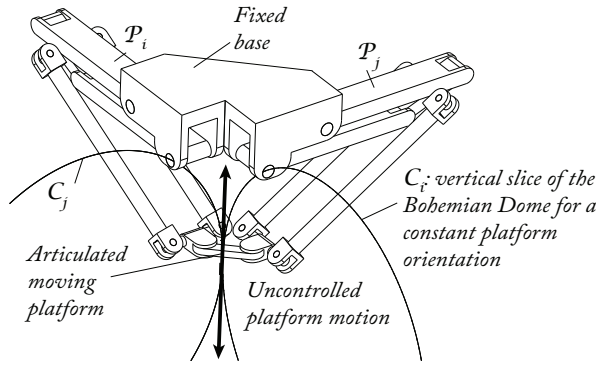


Figure 3.14 – Example of a Type 2 singularity for a 2-II(2-UU) robot: the platform gets an uncontrollable translation.

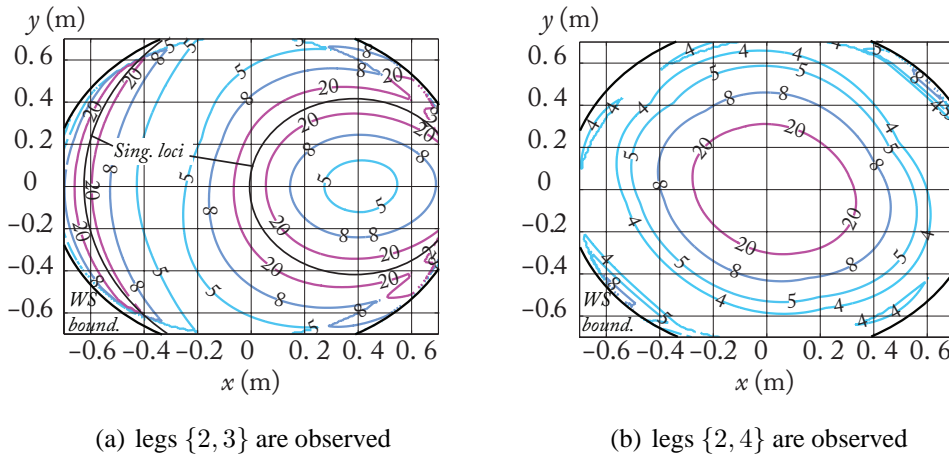


Figure 3.15 – Maximal position error (in mm) for  $z = -0.7$  m and  $\phi = 0$  deg.

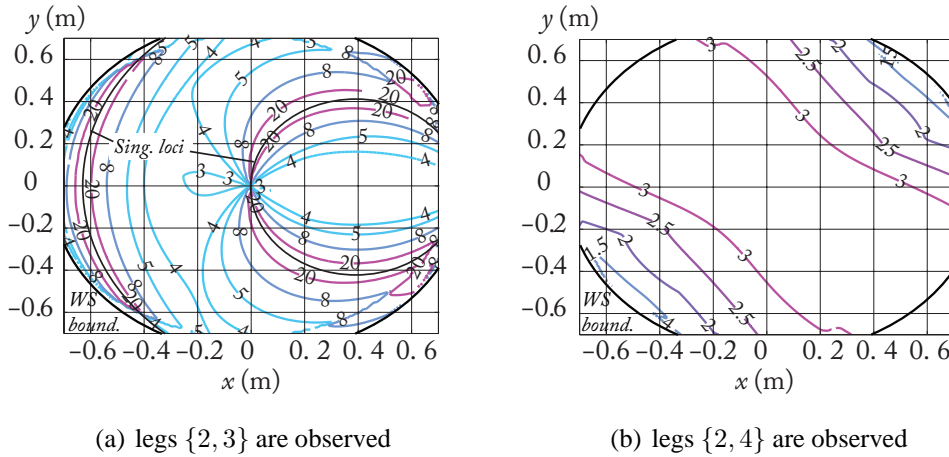


Figure 3.16 – Maximal orientation error (in deg) for  $z = -0.7$  m and  $\phi = 0$  deg.

middle of the workspace only. Thus, it can be concluded that the selection of the legs to observe is crucial for the final pose accuracy.

Let us now compute the maximal positioning error when the four legs are observed. It can be observed that the position error is larger near  $\{x = 0m, y = 0m, z = -0.61m, \phi = 0deg\}$ . This can be explained by the fact that this configuration is a singularity of the model for which all the planes  $P_i$  ( $i = 1, 2, 3, 4$ ) are parallel. Thus, even if all the legs are observed, singular configurations may appear near which the accuracy is poor. Such a phenomenon shows the

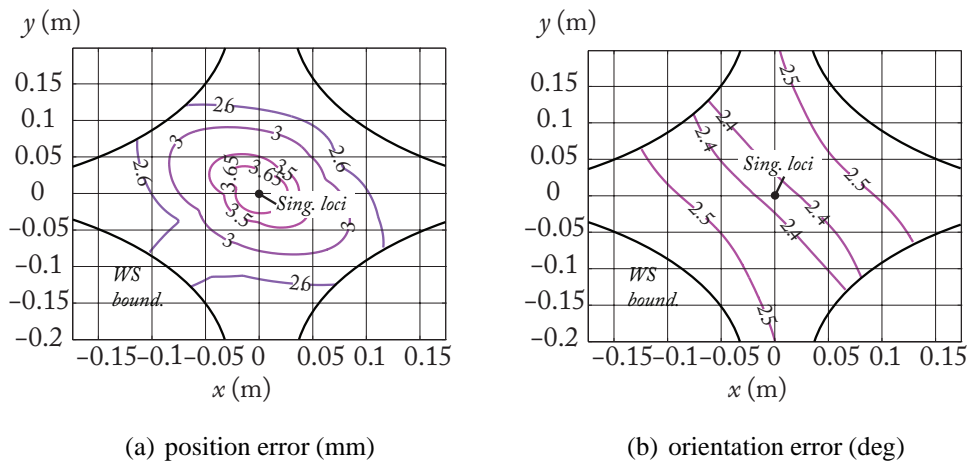


Figure 3.17 – Maximal position and orientation error for  $z = -0.61$  m and  $\phi = 0$  deg when all legs are observed.

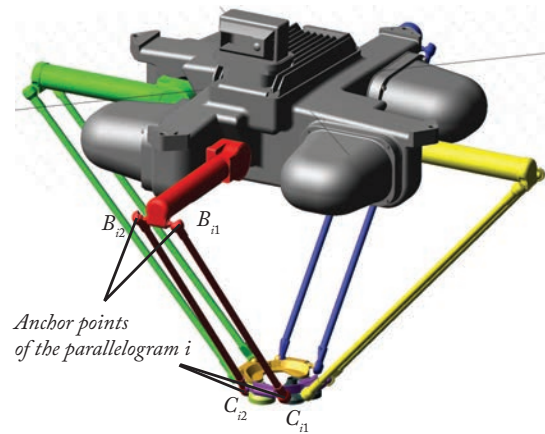


Figure 3.18 – The ADAMS mockup of the Adept Quattro connected to Matlab/Simulink via the module ADAMS/Controls

importance of the study of the intrinsic properties of the controller via the hidden robot concept.

### 3.3.2 Simulation Results

#### 3.3.2.1 Description of the simulator

The simulations are performed on an ADAMS mockup of the Adept Quattro (Figure 3.18) with the same kinematic properties as the real robot by Adept. This virtual mockup is connected to Matlab/Simulink via the ADAMS/Controls module. The controller presented in Section 2.3.2 is applied (with  $\lambda = 0.8$  – Figure 3.19) in which:

- the observation of the leg is simulated by extracting in the ADAMS model the positions of the anchor points of each parallelograms,
- the actuated joint velocities are given as inputs of the ADAMS mockup.

On the used control scheme, we can also switch on or off simulated measurement noise (whose amplitude can be parametrised) on the observation of the leg directions.

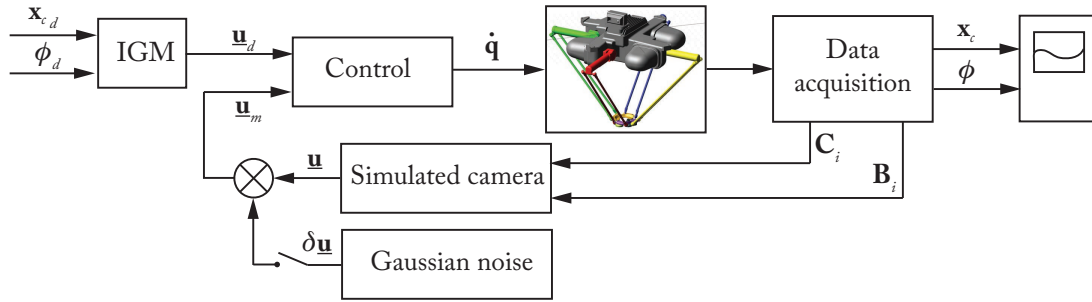


Figure 3.19 – The controller used for the simulations

### 3.3.2.2 Numerical validations

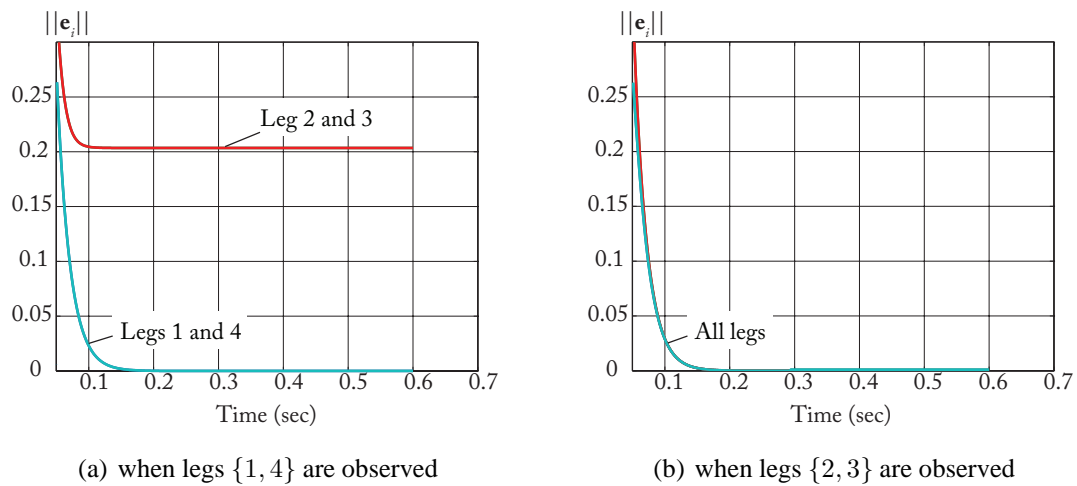
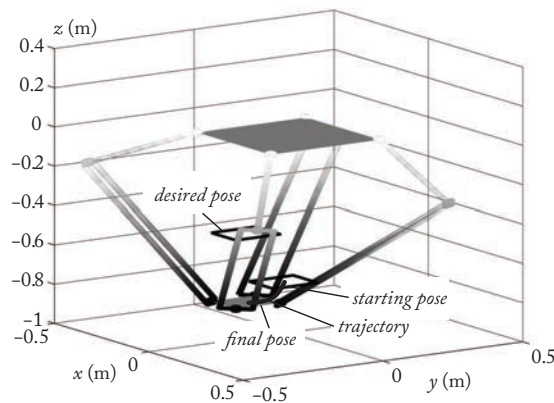
**Testing the Convergence of the Robot to the Desired Pose** In the first simulation, no noise is added on the simulated values of the leg directions. The initial platform pose is equal to  $\{x = 0m, y = 0m, z = -0.75m, \phi = 0deg\}$  and the final desired platform pose is set to  $\{x = -0.2m, y = 0m, z = -0.56m, \phi = 0deg\}$ . For going from the initial point to the final ones, two sets of observed leg directions are tested:  $\{1, 4\}$  and  $\{2, 3\}$ . For those two set of legs, solving the *fkp* of the hidden robot model of the Quattro presented in Section 4.2.2 at the desired final configuration of the robot, the following assembly modes can be obtained:

- for legs  $\{1, 4\}$ :
  - solution 1:  $\{x = -0.2m, y = 0m, z = -0.56m, \phi = 0deg\}$
  - solution 2:  $\{x = -0.2m, y = 0m, z = -0.909m, \phi = 0deg\}$
  - solution 3:  $\{x = -0.138m, y = 0.062m, z = -1.019m, \phi = 0deg\}$
  - solution 4:  $\{x = -0.138m, y = 0.062m, z = -0.45m, \phi = 0deg\}$
- for legs  $\{2, 3\}$ :
  - solution 1:  $\{x = -0.2m, y = 0m, z = -0.56m, \phi = 0deg\}$
  - solution 2:  $\{x = -0.2m, y = 0m, z = -0.296m, \phi = 0deg\}$
  - solution 3:  $\{x = -0.262m, y = 0.062m, z = -0.694m, \phi = 0deg\}$
  - solution 4:  $\{x = -0.262m, y = 0.062m, z = -0.161m, \phi = 0deg\}$

The results for the convergence of the leg directions are presented in Figure 3.20. It can be shown that when the legs  $\{2, 3\}$  are observed, all leg directions converge to 0. This is not true for the second case. Looking at the platform pose computed by ADAMS, the robot reach the configuration  $\{x = -0.2m, y = m, z = -0.909m, \phi = 0deg\}$ , i.e. the second solution for the *fkp* of the hidden robot model of the Quattro (Figure 3.21).

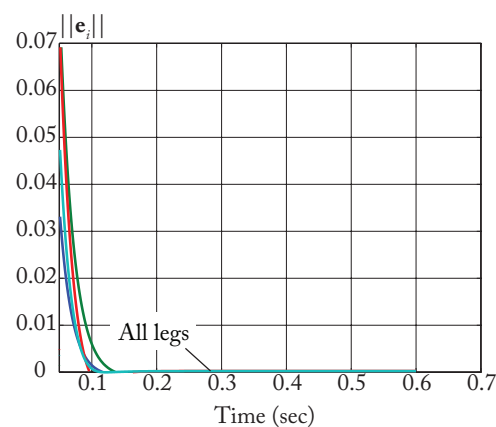
A second simulation is performed in which all legs are observed. The initial platform pose is equal to  $\{x = 0.05m, y = 0.05m, z = -0.8m, \phi = 0deg\}$  and the final desired platform pose is set to  $\{x = 0.03m, y = 0.03m, z = -0.59m, \phi = 0deg\}$ . Solving the *fkp* of the hidden robot model of the Quattro when all legs are observed at the desired final configuration of the robot, it can be proven that there still exist two assembly modes which are:

- solution 1:  $\{x = 0.03m, y = 0.03m, z = -0.59m, \phi = 0deg\}$

Figure 3.20 – Error norm on each leg  $\|e_i\|$ .Figure 3.21 – Initial, desired and final position when legs  $\{1, 4\}$  are observed

— solution 2:  $\{x = 0.03m, y = 0.03m, z = -0.65m, \phi = 0deg\}$

Looking at the platform pose computed by ADAMS, even if all errors on the legs vanish (Figure 3.22), the robot reaches the configuration  $\{x = 0.03m, y = 0.03m, z = -0.65m, \phi = 0deg\}$ , i.e. the second solution for the *fkp* (Figure 3.23).

Figure 3.22 – Error norm on each leg  $\|e_i\|$  when all the legs are observed.



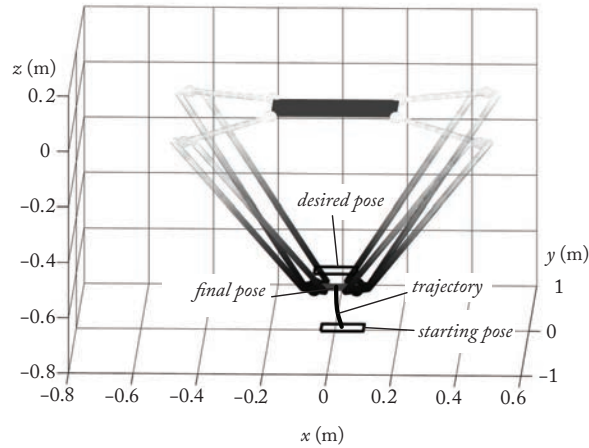


Figure 3.23 – Initial, desired and final position when all legs are observed

All these numerical results confirm the presence of the virtual robot hidden within the controller that must be studied in order to avoid convergence problems due to inadequate stacking of interaction matrices.

**Testing the Presence of Local Minima** For that simulations, all legs are observed. The initial platform pose is equal to  $\{x = 0.028m, y = 0m, z = -0.617m, \phi = 0deg\}$  and the final desired platform pose is set to  $\{x = -0.6m, y = 0m, z = -0.8m, \phi = 0deg\}$ . No noise is added on the simulated values of the leg directions. After about 0.3 s of simulations, the robot stops in the configuration  $\{x = -0.588m, y = 0m, z = -0.847m, \phi = 0deg\}$  while the error on the leg direction is far from zero (Figure 3.24). Thus we are in the presence of a local minimum.

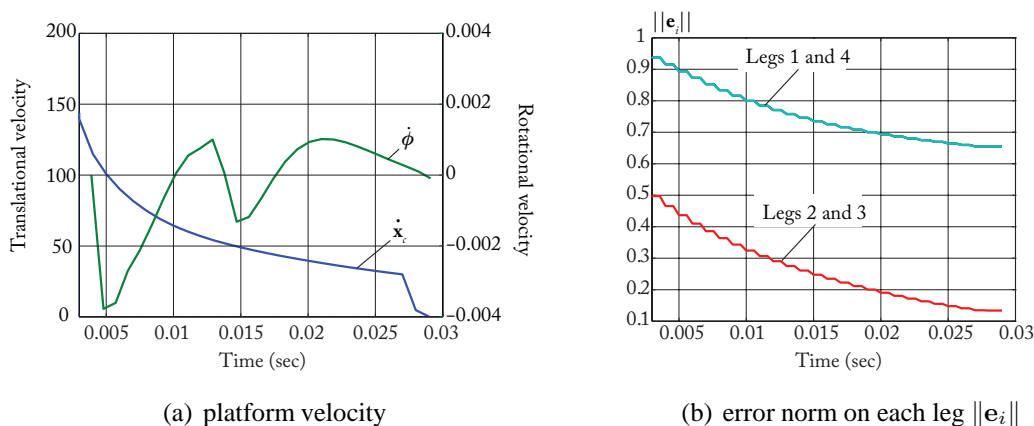


Figure 3.24 – Platform velocity and error norm on each leg when the robot meets a local minimum.

Looking at the configuration in which the robot is blocked, it appears that, as forecast, it is a Type 1 singularity (boundary of the workspace (Figure 3.25)). This confirms the fact that the local minima appear in the Type 1 singularities of the hidden robot model, as mentioned in Section 3.2.3.



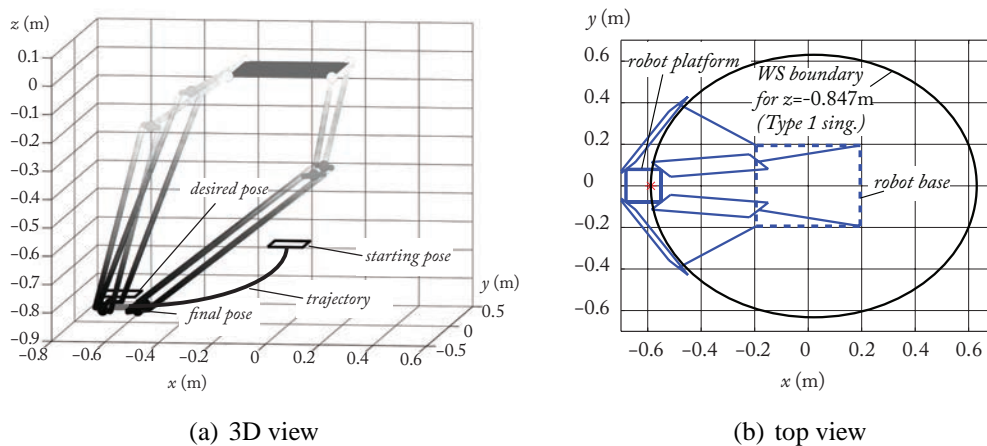


Figure 3.25 – Robot configuration when it meets a local minimum.

### Testing the Importance of the Selection of the Observed Legs on the Robot Accuracy

In the first simulation, the initial platform pose is equal to  $\{x = 0.02m, y = 0.1m, z = -0.7m, \phi = 0deg\}$  and the final desired platform pose is set to  $\{x = -0.2m, y = 0.01m, z = -0.7m, \phi = 0deg\}$ . A random noise of 0.1 deg is added to the simulated measure of the leg directions. To show the importance of the leg selection on the robot accuracy, it is decided to control the robot displacement using two different sets of legs: (i) legs  $\{2, 3\}$  and (ii) legs  $\{2, 4\}$ . The results (Figure 3.26) show that, as presented in Figure 3.15, the final platform pose accuracy is better when legs  $\{2, 3\}$  are observed (around 3 mm and 0.05 rad) than with legs  $\{2, 4\}$  (around 7 mm and 0.07 rad).

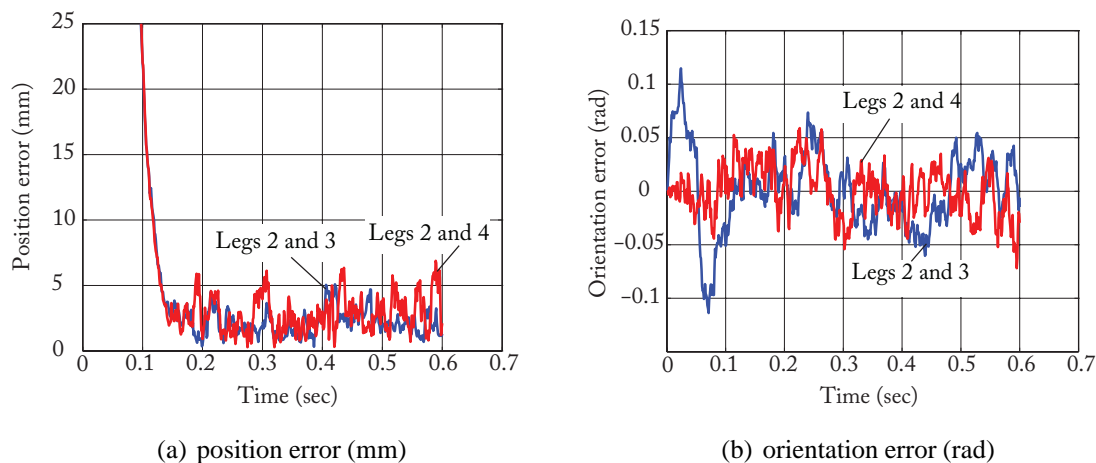


Figure 3.26 – Orientation and position error when legs  $\{2, 3\}$  and legs  $\{2, 4\}$  are observed.

In the second simulation, the initial platform pose is equal to  $\{x = 0.05m, y = 0.05m, z = -0.8m, \phi = 0deg\}$  and the final desired platform pose is set to  $\{x = 0.03m, y = 0.03m, z = -0.65m, \phi = 0deg\}$ . It is decided to control the robot displacement using three different sets of legs: (i) legs  $\{1, 4\}$ , (ii) legs  $\{1, 3, 4\}$  and (iii) all legs. The results (Figure 3.27) show that the final platform pose accuracy is better when all legs are observed, while the accuracy is quite the same when two or three legs are observed. However, this result must not hide the fact that, even if four legs can lead to better accuracy, some convergence problems can still appear, as shown

previously.

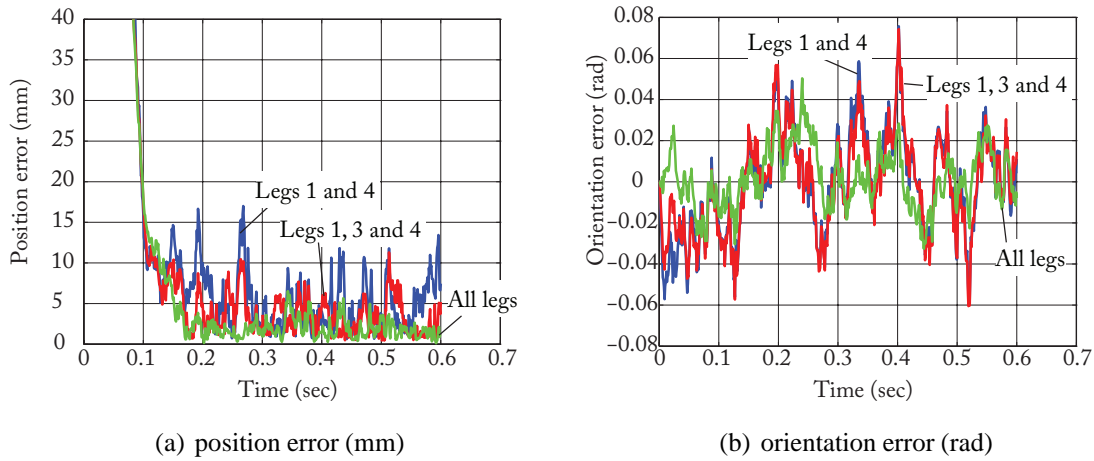


Figure 3.27 – Orientation and position error when legs  $\{1, 4\}$ ,  $\{1, 3, 4\}$  and all legs are observed.

### 3.3.3 Experimental Results

#### 3.3.3.1 Description of the benchmark

In this section, experiments are now performed on a real Adept Quattro. The benchmark is composed of (Figure 3.28):

- an Adept Quattro robot bought by the Institut Pascal of Clermont-Ferrand (France),
- a camera AVT Marlin F131B firewire IEEE1394 (lens: 3.6mm 1:1.6 1/2 inch for CCD camera), which is mounted at the centre of the robot base so that all the legs can be observed without any problems of occlusion and whose intrinsic and extrinsic parameters have been calibrated,
- a lighting system that provides an homogenous lighting to the scene,
- a computer that extracts the data coming from the camera, computes the value of the leg directions  $\mathbf{u}$ , then calculates the robot actuator velocity  $\dot{\mathbf{q}}$  using the controller of Section 2.3.2 and send the information to the robot controller. Note that, in experiments, the value of  $\lambda$  in the controller is fixed to 0.2.

Moreover, the robot is covered by a cloth that prevents the lighting variations and guarantees the contrast quality required for observing the black legs of the robot (Figure 3.29).

Finally, it must be mentioned we have deliberately decided to use the minimal camera resolution and to not undistort the image captured. The measurement noise on the leg direction is thus of about 0.1 rad, but:

- such a high noise is interesting to show the controller robustness to leg direction prediction errors,
- the noise is so high that, for analyzing the robot accuracy and measuring the distance between the real and nominal robot configurations, we can directly record and use the

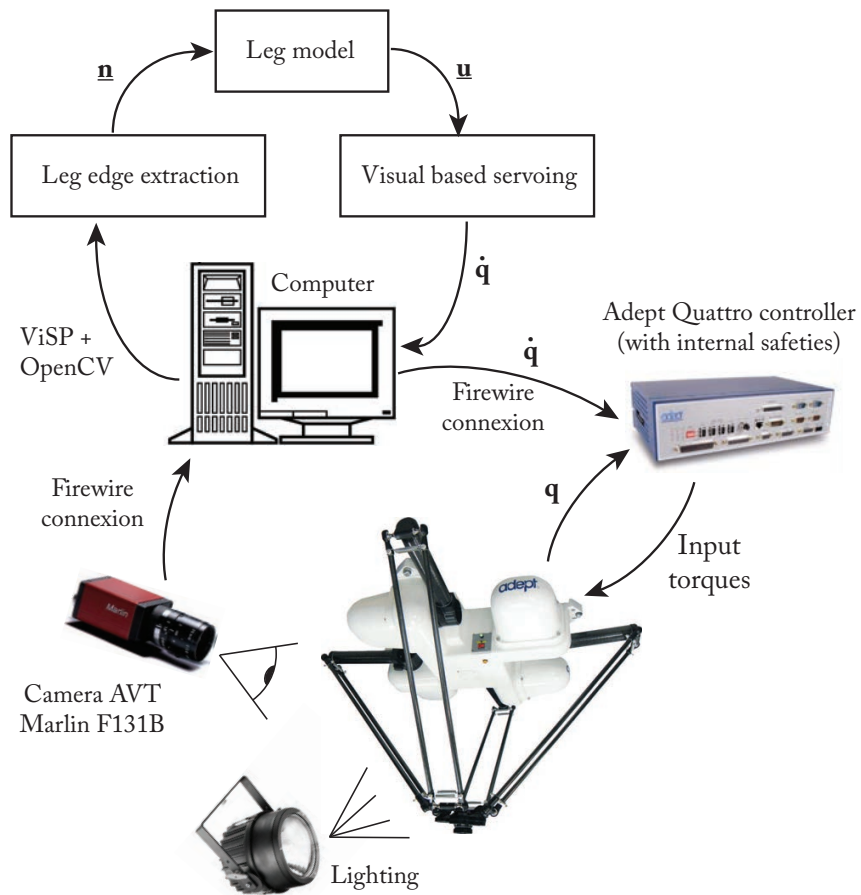


Figure 3.28 – Experimental bench

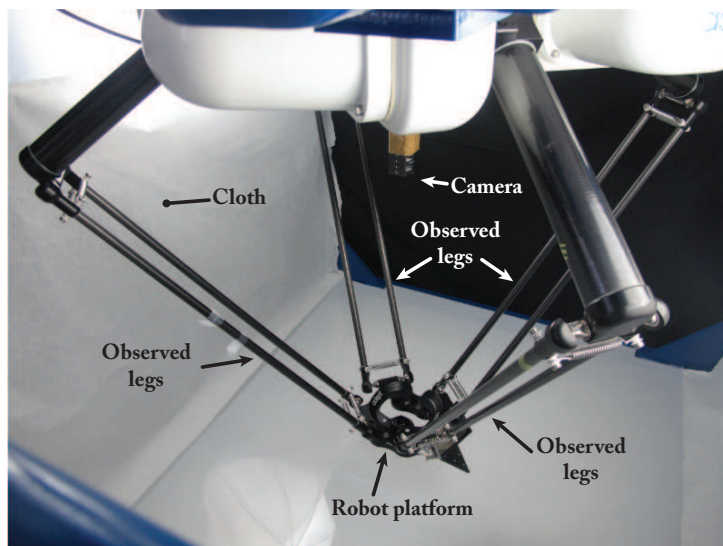


Figure 3.29 – The Quattro recovered by the cloth

value of the platform pose predicted by the Adept Quattro controller instead of using one external measurement device (such as a lasertracker).

### 3.3.3.2 Experimental validations

**Testing the Convergence of the Robot to the Desired Pose** We replay now experimentally the convergence tests presented in Section 3.3.2. The starting and desired final points are the same as previously. The results are presented in the Tables 3.2 to 3.4 and illustrated by the Figs. 3.30 to 3.32. It should be mentioned that, for cross-validating the results on those pictures, the plotted values of the error norms are computed using the values of the leg directions given by the Quattro controller.

Due to the presence of high measurement noise, the robot can of course not converge to the final desired pose. Therefore, in these Tables, information on the tolerable maximal error on the pose attained in simulations is given. Please note that, due to the large value of the error on the measured angle, the model defined in Section 3.3.1 is no longer valuable and we have preferred to use a more refined non linearised model proposed in (Briot and Boney, 2010).

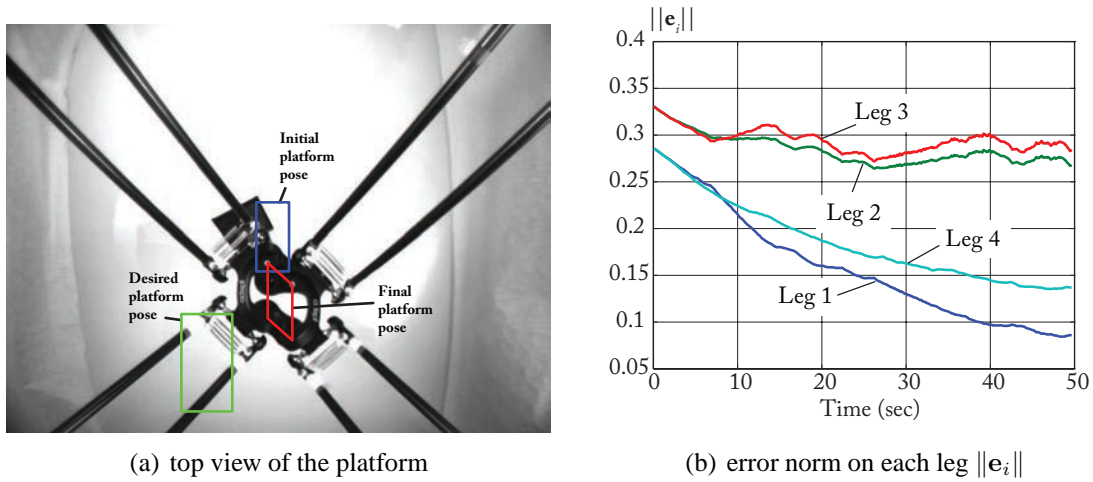


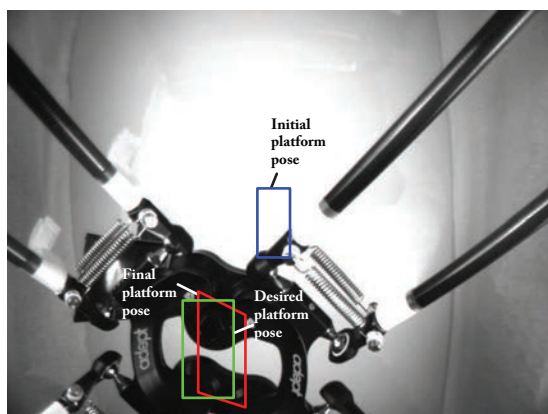
Figure 3.30 – Convergence of the robot when legs 1 and 4 are observed (desired pose:  $\{x = -0.2, y = 0, z = -0.56, \phi = 0\}$ ).

Table 3.2 – Results on the experiments carried out for testing the convergence of the robot when legs 1 and 4 are observed (the positions are in meters, the angles in radians).

Desired final pose	$\{x = -0.2, y = 0, z = -0.56, \phi = 0\}$
Final pose in simulation	$\{x = -0.2, y = 0, z = -0.91, \phi = 0\}$
Tolerable position error	0.11 m
Tolerable orientation error	2.00 rad
Final pose in experiments	$\{x = -0.11, y = 0.01, z = -0.86, \phi = -2.15\}$
Distance to the final pose in simulation	0.10 m
Orient. err. w.r.t. the final pose in simulation	2.15 rad

All these experimental results match with the simulation results presented above and confirm the presence of the virtual robot hidden within the controller that must be studied in order to avoid the convergence problems due to inadequate stacking of interaction matrices.

**Testing the Presence of Local Minima** Unfortunately, we were not able to do such experiments as the robot controller is designed with safeties that cannot be suppressed and that prevent



(a) top view of the platform

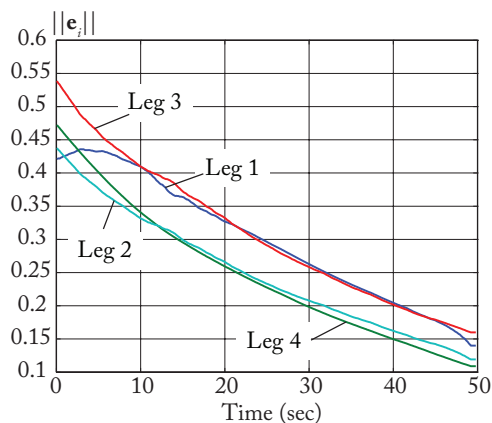
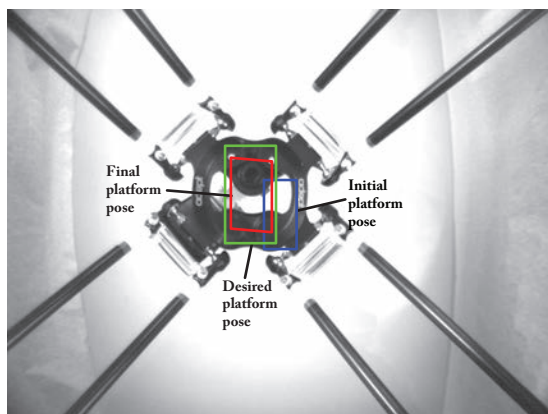
(b) error norm on each leg  $\|e_i\|$ 

Figure 3.31 – Convergence of the robot when legs 2 and 3 are observed (desired pose:  $\{x = -0.2, y = 0, z = -0.56, \phi = 0\}$ ).

Table 3.3 – Results on the experiments carried out for testing the convergence of the robot when legs 2 and 3 are observed (the positions are in meters, the angles in radians).

Desired final pose	$\{x = -0.2, y = 0, z = -0.56, \phi = 0\}$
Final pose in simulation	$\{x = -0.2, y = 0, z = -0.56, \phi = 0\}$
Tolerable position error	0.23 m
Tolerable orientation error	1.23 rad
Final pose in experiments	$\{x = -0.12, y = 0.05, z = -0.55, \phi = -0.90\}$
Distance to the final pose in simulation	0.10 m
Orient. err. w.r.t. the final pose in simulation	0.90 rad



(a) top view of the platform

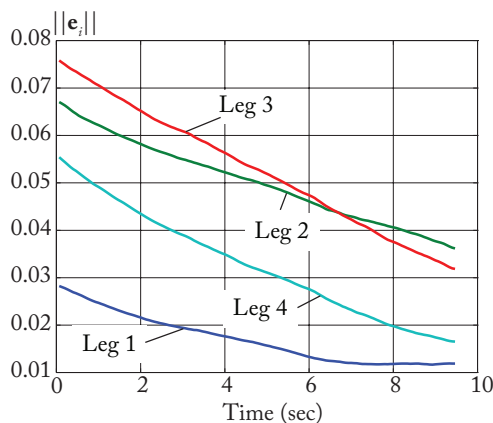
(b) error norm on each leg  $\|e_i\|$ 

Figure 3.32 – Convergence of the robot when all legs are observed (desired pose:  $\{x = 0.03, y = 0.03, z = -0.59, \phi = 0\}$ ).

going into singularities. However, as the presence of local minima that are located in the Type 1 singularities was demonstrated in simulations, we think that this numerical proof brings enough strength to our demonstration concerning this point.

**Testing the Importance of the Selection of the Observed Legs on the Robot Accuracy** We replay now experimentally the accuracy tests presented in Section 3.3.2. The starting and de-

Table 3.4 – Results on the experiments carried out for testing the convergence of the robot all legs are observed (the positions are in meters, the angles in radians).

Desired final pose	$\{x = 0.03, y = 0.03, z = -0.59, \phi = 0\}$
Final pose in simulation	$\{x = 0.03, y = 0.03, z = -0.65, \phi = 0\}$
Tolerable position error	0.08 m
Tolerable orientation error	1.54 rad
Final pose in experiments	$\{x = 0.05, y = 0.03, z = -0.72, \phi = 0.05\}$
Distance to the final pose in simulation	0.07 m
Orient. err. w.r.t. the final pose in simulation	0.05 rad

sired final points are the same as previously, as well as the observed legs. Each experiment is run five times and we present here the maximal values obtained on the position and orientation error. The results are shown in the Tables 3.5 to 3.6.

Table 3.5 – Results on the experiments carried out for testing the accuracy of the robot when legs  $\{2, 3\}$  or  $\{2, 4\}$  are observed.

Desired final pose	$\{x = -0.2m, y = 0.01m, z = -0.7m, \phi = 0deg\}$	
Legs	$\{2, 3\}$	$\{2, 4\}$
Position error	0.11 m	0.23 m
Orientation error	0.06 rad	0.68 rad

Table 3.6 – Results on the experiments carried out for testing the accuracy of the robot when legs  $\{1, 4\}$ ,  $\{1, 3, 4\}$  or  $\{1, 2, 3, 4\}$  are observed.

Desired final pose	$\{x = 0.03m, y = 0.03m, z = -0.65m, \phi = 0deg\}$		
Legs	$\{1, 4\}$	$\{1, 3, 4\}$	$\{1, 2, 3, 4\}$
Position error	0.11 m	0.09 m	0.07 m
Orientation error	0.39 rad	0.31 rad	0.05 rad

Once again, all these experimental results match with the simulation results presented above and confirm the necessity to carefully select the set of legs to observe in order to obtain the best accuracy possible. However, it must be recalled that, even if observing all the legs lead to a better accuracy, this result must not hide the fact that some convergence problems can still appear, as shown previously.

### 3.4 Conclusions

This chapter presents the hidden robot concept, a tangible visualisation of the mapping between the observation space and Cartesian space of parallel robots controlled through the use of leg direction-based visual servoing.

The architecture of the hidden robot is explored, noting how it differs from that of the real robot. Certain properties are established for the hidden robot legs, in terms of kinematic behaviour, which are then used to define a general methodology for obtaining the hidden robot leg.



The hidden robot concept is generalised for any parallel robot, and using the aforementioned methodology, it is possible to obtain the associated hidden robot for any parallel kinematic manipulator.

Due to the differences in architecture and kinematics between the hidden robot and the real robot, the former presents different assembly modes and singularities from the latter. This is used to explain certain behaviour noticed when using leg direction-based visual servoing. Using the hidden robot as a tool, it is possible to obtain a few immediate results regarding the convergence and possible non-convergence of the platform to the desired pose, as well as information about singularities within the mapping.

Then, the hidden robot concept is applied to an Adept Quattro robot, and is used to predict in simulation the behaviour regarding convergence and non-convergence to the desired pose with respect to the number of observed legs, as well as accuracy based on the selection of observed legs and studies of local minima. All simulation-based predictions are validated experimentally.

These were all immediate results of using the hidden robot concept to characterise the mapping introduced by the observation of the leg directions. In the following chapter, further applications are presented, including the possibility of analysing the controllability of different robot families using geometric methods (due to the existence of the hidden robot), algorithms for selecting the optimal set of legs to be observed, and introducing additional information to make previously uncontrollable robots partially or even fully controllable.







# 4

---

## Extending the Applications of the Hidden Robot Concept

*This chapter expands on the applications of the hidden robot concept. First, using the methodology described in the previous chapter, the hidden robot is constructed and its assembly modes and singularities identified for different classes of planar and spatial parallel robots. Then, a methodology is proposed for the optimal leg selection to be observed using different algorithms which are then applied in simulations on the Adept Quattro presented earlier. The hidden robot is further used in the controllability analysis using leg-based observation, which is then illustrated on a special type of 3-PRR manipulator. Finally, for robots which cannot be fully controlled using leg direction-based observation, alternative feature observation is presented, which leads to different hidden robot architectures.*

### 4.1 Analysis of the Hidden Robots of Planar Manipulators

#### 4.1.1 The Hidden Robot Legs of Planar Parallel Robots

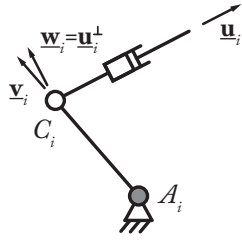
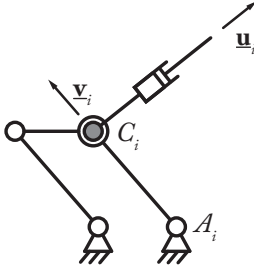
The usual planar parallel manipulators (*ppm*) are composed of planar serial chains with at most three 1-*dof* joints, respectively, among which one is actuated. As mentioned in (Merlet, 2006b), using the different possible combinations of R and P joints, only 10 different serial chains, that lead to robots that can be actuated, can be obtained. These chains are represented in Table 4.1 (in this table and the following pictures, the gray pairs denote the actuated joints).

Now, using the approach presented in Section 3.2.2, and considering that the direction  $\mathbf{u}_i$  of the last segment of each leg is observed, one can find the hidden robot leg corresponding to this observation (Table 4.1).

Table 4.1: The 10 possible architectures for the legs of  $ppm$  and their equivalent hidden robot leg for the visual servoing using leg directions

Real leg architecture	Hidden robot leg architecture
<p>for <math>\underline{R}RR</math> and <math>R\underline{R}R</math> legs <math>\implies</math></p>	<p>a <math>\Pi RR</math> leg</p>
<p>for <math>\underline{R}PR</math> and <math>R\underline{P}R</math> legs <math>\implies</math></p>	<p>a <math>\underline{R}PR</math> leg</p>
<p>for <math>\underline{P}RR</math> and <math>P\underline{R}R</math> legs <math>\implies</math></p>	<p>a <math>\underline{P}RR</math> leg</p>
<p>for <math>\underline{P}PR</math> and <math>P\underline{P}R</math> legs <math>\implies</math></p>	<p>no equivalent hidden robot leg</p>
<p>for <math>\underline{P}RP</math> legs <math>\implies</math></p>	<p>a <math>\underline{P}RP</math> leg</p>

Table 4.1: The 10 possible architectures for the legs of  $ppm$  and their equivalent hidden robot leg for the visual servoing using leg directions

Real leg architecture	Hidden robot leg architecture
<p>for <math>\underline{RRP}</math> legs <math>\implies</math></p> 	<p>a <math>\Pi\underline{RRP}</math> leg</p> 

From Table 4.1, the following information can be extracted:

- for  $\underline{RPR}$  and  $\underline{PRR}$  legs, the hidden robot legs are the same as the real ones;
- $\underline{PRP}$  legs lead to  $\underline{PRP}$  hidden robot legs; and robots made of  $\underline{PRP}$  legs are well known not to be controllable (Merlet, 2006b). A similar result appears for  $\underline{RRP}$  legs that lead to  $\underline{\Pi RRP}$  hidden robot legs.
- the last element of  $\underline{PPR}$  and  $\underline{PPR}$  having a constant direction  $\underline{u}_i$ , robots made of such legs cannot be controlled using leg direction observation. As a result, they don't have an equivalent hidden robot model.

Thus, using the concept of hidden robot leg and hidden robot model, the problem of the robot controllability can be directly addressed without any mathematical derivations.

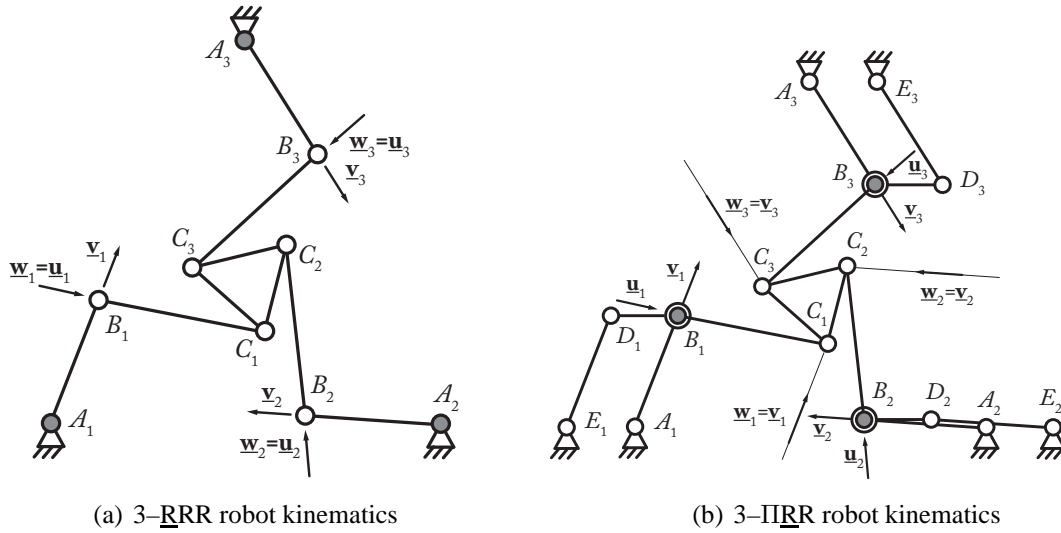
The next Section presents the hidden robot models of the 2 and 3-*dof* controllable robots with a symmetric leg arrangement and made of the legs presented in Table 4.1.

### 4.1.2 The Hidden Robot Models of Planar Parallel Robots

Using the results of the previous Section, 12  $ppm$  with symmetric leg arrangement (i.e. with identical leg architectures) that can be controlled using the leg direction observation can be found:

- for manipulators with 2 *dof*:  $\underline{RRRRR}$ ,  $\underline{RRRRR}$ ,  $\underline{RPRPR}$ ,  $\underline{RPRPR}$ ,  $\underline{PRRRP}$ ,  $\underline{PRRRP}$  robots;
- for manipulators with 3 *dof*: 3- $\underline{RRR}$ , 3- $\underline{RRR}$ , 3- $\underline{RPR}$ , 3- $\underline{RPR}$ , 3- $\underline{PRR}$ , 3- $\underline{PRR}$  robots.

Their architectures are well-known and, for the reason of writing clarity and as they can also easily be found by the assembly of the legs presented in Table 4.1, their schematics, as well as the corresponding architecture of their hidden robot models, are only detailed in the Appendix (Tables A.1 and A.2).

Figure 4.1 – The 3-RRR robot and its hidden robot model

For illustrating this Section, let us present the (*fkp*) and singularity analysis of the hidden robot model of the 3-RRR robot, when controlled using leg direction observation (Figure 4.1(a)). Using the results of Table 4.1, it can be found that its equivalent hidden robot model is a 3-IIRR robot (Figure 4.1(b)). Each of its legs is composed of a passive planar parallelogram ( $\Pi$  joint) which is able to maintain constant the orientation of the links  $B_iD_i$  with respect to the base and of an RR chain which is mounted on the link  $B_iD_i$ .

**Forward Kinematics and Assembly Modes** Using the usual methodology (Merlet, 2006b), all the solutions to the *fkp* are at the intersections of the coupler curve (which represents the displacement loci of one platform extremity when one of the leg is disassembled, the actuators of the two other being fixed (see Figure 4.2(a))) with the vertex space of the disassembled leg (that represents the passive displacement of the leg tip when the actuator is fixed (see Table 4.1)). For the studied 3-IIRR robot, as the leg vertex spaces are circles (Table 4.1), the coupler curve is a sextic curve (Merlet, 2006b), i.e. an algebraic curve of degree 6 (in the case where the vertex spaces had been lines, the coupler curve would have been an ellipse (Briot et al., 2008)). Thus, the solutions of the *fkp* are at the intersection points between the aforementioned circle and sextic curve. And it is shown in (Merlet, 2006b) that if a circle intersects a sextic curve, there are at most 6 intersection points. An example of possible assembly modes for the 3-IIRR robot are presented in Figure 4.2(b).

It should be mentioned that, even if for the 3-RRR (and as a consequence, for the 3-RRR), the hidden robot has 6 assembly modes, for the other 10 robots cited at the beginning of this Section, the maximal number of assembly modes is 2 (see Appendix), i.e. the control approach based on the observation of the leg direction allows most of the time the decrease of complexity for the *fkp*.

**Singular Configurations** The Type 2 singular configurations of *ppm* have been deeply studied in the past and are well-known. For 3-*dof ppm* (moving in the  $Oxy$  plane), the singularities

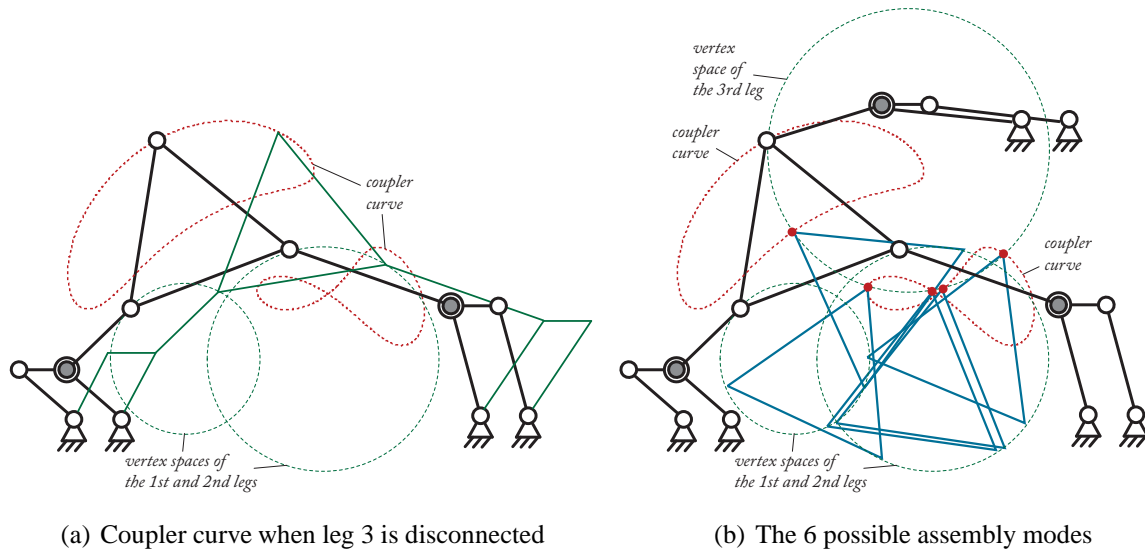
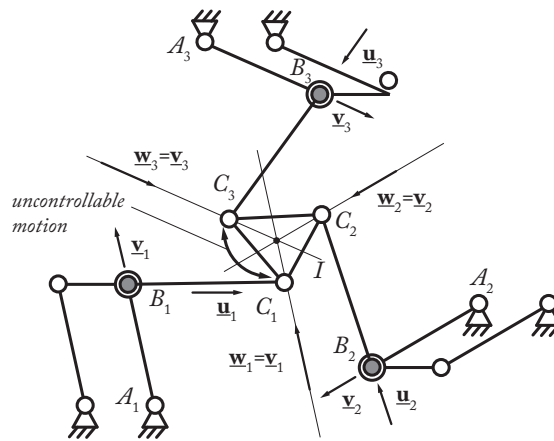
Figure 4.2 – Solutions of the  $fkp$  for a 3-PIRR robot

Figure 4.3 – Example of Type 2 singularities for the 3-PIRR robot

appear when  $\mathbf{s}_1 \cdot (\mathbf{s}_2 \times \mathbf{s}_3) = 0$ , where  $\mathbf{s}_i^T = [w_j^x \ w_j^y \ m_j^z]^T$  in which  $w_j^x$  and  $w_j^y$  are the  $x$  and  $y$  components of  $\mathbf{w}_j$  ( $\mathbf{w}_j$  corresponds to the direction of the effort applied by the actuated leg on the platform (Bonev et al., 2003) – see Table 4.1 and Figure 4.1) and  $m_j^z$  is the moment of  $[w_j^x \ w_j^y]^T$  (note that  $\mathbf{w}_j$  is always applied at point  $C_j$ ) (Bonev et al., 2003). Such a condition means that the lines of Plücker coordinates  $\mathbf{s}_i$  intersect in one single point  $I$  that corresponds to the instantaneous centre of rotation of the platform in the uncontrolled  $dof$  (Figure 4.3). This point can be at infinity: in this case, all vectors  $\mathbf{w}_j$  are parallel and the robot gets one uncontrolled translational motion in the direction orthogonal to  $\mathbf{w}_j$ .

Obtained results show that singular configurations of the 3-PIRR robot are different from the singular configurations of the real 3-RRR robot, which appear when all lines passing through  $C_i$  of direction  $\mathbf{u}_i$  intersect in one point (Bonev et al., 2003).

Let us now apply the concept of hidden robot models to some particular classes of spatial parallel robots.

## 4.2 Analysis of the Hidden Robots of Spatial Manipulators

For spatial parallel robots, due to the existence of hundreds of possible different architectures and due to the difficulty of classifying the robots by families, it is not possible to present all the hidden robot models. Thus, it is decided in this Section to show the hidden robot models of two of the best known families of parallel robots:

- the  $n$ -Pod family (i.e. robots such as Hexapods (or GS platforms) (Gough and Whitehall, 1962), the Tsai mechanism (Tsai, 2000), etc.)
- the Delta-like robot family (i.e. robots such as the Delta (Clavel, 1990), the Quattro (Nabat et al., 2005), the Orthoglide (Chablat and Wenger, 2003), etc.)

For other types of robot architectures, the methodology described in Section 3.2.2 can be applied for finding a possible hidden robot model, whose singularity configuration problem can then be studied using known Grassmann Geometry and Grassmann-Cayley Algebra methods (Merlet, 2006b; Ben-Horin and Shoham, 2006; Caro et al., 2010b).

### 4.2.1 The $n$ -Pod Robot Family

The  $n$ -Pod robot family regroups the robots made of  $n$   $\underline{U}\underline{P}\underline{S}$  legs (Figure ??), or some of their variations such as  $\underline{U}\underline{P}\underline{U}$  (Tsai, 2000) or even  $\underline{R}\underline{P}\underline{S}$  legs (Bonev, 2008), i.e. the legs are composed of one passive  $\underline{U}$  or  $\underline{R}$  joint located on the ground, followed by an active  $\underline{P}$  joint and then by one passive  $\underline{S}$  or  $\underline{U}$  joint.

Probably the most known robots of this family are the GS platform (Gough and Whitehall, 1962) (Figure 2.14), the 3- $\underline{U}\underline{P}\underline{U}$  robots (e.g. see (Tsai, 2000; Wolf et al., 2002)) and the 3- $\underline{R}\underline{P}\underline{S}$  robot (Bonev, 2008).

For such types of legs, the prismatic joint direction can be observed (Andreff et al., 2007; Briot and Martinet, 2013). From Section 3.2.2 and also from (Briot and Martinet, 2013), it can be shown that the virtual equivalent legs are:

- for  $\underline{U}\underline{P}\underline{S}$  legs: a  $\underline{U}\underline{P}\underline{S}$  leg;
- for  $\underline{U}\underline{P}\underline{U}$  legs: a  $\underline{U}\underline{P}\underline{U}$  leg;
- for  $\underline{R}\underline{P}\underline{S}$  legs: a  $\underline{R}\underline{P}\underline{S}$  leg;

i.e. the joint fixed on the ground becomes actuated, while the prismatic joint becomes passive. Then, using the usual methodology, the  $fkp$  and singularity analysis can be carried out.

An illustration of this has been presented on the GS platform in Section 3.1, presenting the analysis of the forward kinematics and assembly modes of the hidden robot, as well as its singularities. Since the rest of the robots within this family behave in a similar fashion, no further illustrative example will be given here.

### 4.2.2 The Delta-like Robot Family

The Delta-like robot family regroups the robots made of  $n$   $\underline{A}\{-2-\underline{U}\underline{S}\}$  legs (where  $\underline{A}$  can be either an active  $\underline{R}$  or  $\underline{P}$  joint – see Figure 3.11(a) for an example of  $\underline{R}\{-2-\underline{U}\underline{S}\}$  leg) (Clavel,

1990; Krut et al., 2003), or some of their variations such as  $\underline{A}UU$  (Tsai and Joshi, 2001), or even  $\underline{A}US$  legs (Pierrot et al., 1990; Honegger et al., 1997), i.e. the leg is composed of one active  $\underline{R}$  or  $\underline{P}$  joint located on the ground, followed by either two passive U joints, a spatial parallelogram (2-US loops) or even passive U and S joints.

Probably the most known robots of this family are the Delta (Clavel, 1990), the Quattro (Nabat et al., 2005) (Figure 3.11(b)), the Orthoglide (Chablat and Wenger, 2003), but many other types of architectures exist (see for example (Krut et al., 2003; Company and Pierrot, 2002)). For such types of legs, the distal links direction can be observed (Figure 3.11(a)). From Section 3.2.2, it can be shown that the virtual equivalent legs are:

- for  $\underline{R}$ -{2-US} legs: a  $\Pi$ -{2- $\underline{U}U$ } or a  $\Pi$ -{2- $\underline{U}U$ } leg (Figure ??);
- for  $\underline{R}UU$  legs: a  $\Pi\underline{U}U$  leg;
- for  $\underline{R}US$  legs: a  $\Pi\underline{U}S$  leg;
- for  $\underline{P}$ -{2-US} legs: a  $P$ -{2- $\underline{U}S$ } or a  $P$ -{2- $\underline{U}U$ } leg;
- for  $\underline{P}UU$  legs: a  $P\underline{U}U$  leg;
- for  $\underline{P}US$  legs: a  $P\underline{U}S$  leg;

i.e. the active  $\underline{R}$  joints fixed on the ground are replaced by a passive planar parallelogram joint, while the active  $\underline{P}$  joints become passive and the passive U or  $R\Pi$  joints become active. Then, using the usual methodology, the *fkp* and singularity analysis can be carried out.

The analysis of the Quattro (with 4 *dof*) has been presented in Section 3.3. Thus, for illustrating this section, let us present the *fkp* and singularity analysis of the hidden robot model of the Quattro with 3 *dof*. Note that this latter model is different from the previously studied Quattro, and its kinematic behaviour is equivalent instead to that of a redundant Delta robot when controlled using leg direction observation.

The 3-*dof* Quattro is made of 4  $\underline{R}$ -{2-US} legs, thus its equivalent hidden robot will be made of  $\Pi$ -{2- $\underline{U}S$ } or  $\Pi$ -{2- $\underline{U}U$ } legs. As such hidden robot legs have 2 degrees of actuation (the U joint is fully actuated), only two legs have to be observed for fully controlling the Quattro using leg direction observation. However in this case, if the hidden robot has a 2- $\Pi$ -{2- $\underline{U}S$ } architecture, the platform will have two uncontrolled *dof*. This phenomenon disappears if  $\Pi$ -{2- $\underline{U}U$ } legs are used in the hidden robot model (Figure ??).

**Forward Kinematics and Assembly Modes** Let us start by looking at the loop closure equations of the Quattro with 3 *dof*:

$${}^i\mathbf{C}_i - {}^i\mathbf{B}_i = l_2 {}^i\mathbf{u}_i \quad (4.1)$$

where

$${}^i\mathbf{B}_i = {}^i\mathbf{A}_i + l_1 \begin{bmatrix} \cos q_i & 0 & \sin q_i \end{bmatrix}^T = {}^i\mathbf{A}_i + l_1 {}^i\mathbf{v}_i \quad (4.2)$$

Using the developed form of (4.1) and (4.2), it comes that:

$$\begin{bmatrix} x_{A_i C_i} - l_2 u_i^x \\ y_{A_i C_i} - l_2 u_i^y \\ z_{A_i C_i} - l_2 u_i^z \end{bmatrix} = l_1 \begin{bmatrix} \cos q_i \\ 0 \\ \sin q_i \end{bmatrix} \quad (4.3)$$

where  $\mathbf{u}_i^T = [u_i^x \ u_i^y \ u_i^z]$ . Rearranging the terms of (4.3) and developing the expressions, a set of equations can be obtained (for  $i = 1 \dots 4$ )

$$\begin{aligned} l_1^2 &= ({}^i x - x_{S_i})^2 + ({}^i z - z_{S_i})^2 \\ 0 &= {}^i y - y_{S_i} \end{aligned} \quad (4.4)$$

where  ${}^i \mathbf{S}_i = [x_{S_i} \ y_{S_i} \ z_{S_i}]^T = {}^i \mathbf{A}_i + l_2 {}^i \mathbf{u}_i + {}^i \overrightarrow{C_i P}$  and  ${}^i \mathbf{P} = [{}^i x \ {}^i y \ {}^i z]^T$ . Equations (4.4) are equations of circles, denoted as  $\mathcal{L}_i$ , of radius  $l_1$ , located in planes  $y = y_{S_i}$  and centred in point  $S_i$  of coordinates  ${}^i \mathbf{S}_i$ . These circles represent the vertex space of the tip of the leg  $i$ , when the vector  $\mathbf{u}_i$  is fixed and the planar parallelogram passively moving (Figure 4.4(a)) (Merlet, 2006b). As a result, the forward kinematic problem (fkp) is equivalent to finding the intersection between the circles  $\mathcal{L}_i$  of the observed legs. In conclusion, there may exist 0 solutions, 1 unique solution, 1 double solution (singularity condition), 2 distinct solutions or an infinite number of solutions (if the circles are superposed, which is also a singularity condition) to the *fkp*.

Thus, the 2- $\Pi$ (2- $\underline{U}$ ) robot can have up to two distinct assembly modes that are different from those of the Quattro.

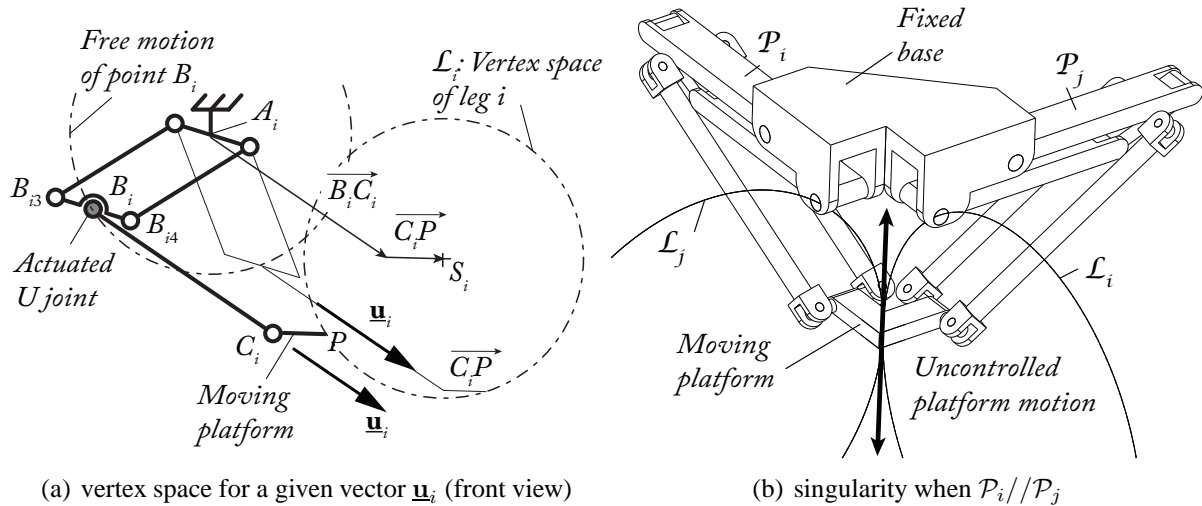


Figure 4.4 – A 2- $\Pi$ (2- $\underline{U}$ ) robot.

**Singular Configurations** For the 2- $\Pi$ (2- $\underline{U}$ ) robot, Type 1 singularities appear when  $\overrightarrow{A_i B_i}$  and  $\overrightarrow{B_i C_i}$  are colinear. In such cases, the robot reaches its workspace boundary. Type 2 singularities appear when the planes  $\mathcal{P}_i$  and  $\mathcal{P}_j$  (whose normal vectors are equal to  $\mathbf{v}_i^\perp$  and  $\mathbf{v}_j^\perp$ , resp.) are parallel. In such cases, the circles  $\mathcal{L}_i$  and  $\mathcal{L}_j$  have a common tangent at their intersection point and the robot gains one uncontrollable *dof* along this tangent (Figure 4.4(b)).



Obviously, the singularity loci vary depending on the leg chosen for the Quattro control. Therefore, it is extremely important, for having the best performances of the controller, to make an optimal selection of the legs to observe. This is the topic of the next section.

## 4.3 Optimal Leg Selection

### 4.3.1 Questions Regarding Leg Selection

Section 3.3 showed the importance of the legs chosen for the control scheme of the Adept Quattro. Several questions naturally arise. The first one concerns the number of legs to observe. In terms of accuracy, it is obvious that observing three or four legs, i.e. adding measurement redundancy, will improve the pose accuracy of the robot. However, increasing the number of legs to observe leads to an increase of the computational time and may be applied with difficulty when high sampling periods are required. Thus, a compromise must be found between the sampling period and the computational time for any given application. Also, observation of all legs might not guarantee a singularity-free mapping, as presented earlier.

The second question is about the selection of the legs to observe. When observing the minimal required number of legs, only two among the four, six different hidden robots can be obtained for the Quattro. From these, two architectures can be eliminated, since they do not allow for the control of the end-effector rotation around the  $z$  axis, due to the way in which this motion is obtained for this particular robot. However, in the case of the GS platform presented in Section 3.1, the minimal required number of legs, only three legs among six, leads to twenty different 3-UPS robots that can be defined. What is thus the best virtual robot model to use?

If the control law proposed at section 2.58 is applied, it is first necessary to guaranty that, for the used set of legs:

- obviously, the legs must be observable during the whole robot displacement,
- the initial and final robot configurations must be included in the same assembly mode of the virtual 3-UPS robot. If not, the controller will not be able to converge to the desired end-effector pose, even if the observed leg directions do. In this last case, the problem can be solved by applying special trajectories that cross Type 2 singularities (Briot and Arakelian, 2008) or encircle a cusp point (Zein et al., 2008).

Finally, we considered that the sensor measurement space is the same as the leg direction space. However, for example using a camera, the leg directions are not directly measured but rebuilt from the observation of the edges of legs projected onto the 2D camera space (Andreff et al., 2005). Thus, for the leg reconstruction, the mapping between the camera space and the real 3D space is involved, and it is not free of singularities (see (Michel and Rives, 1993) for an example of mapping singularities). In the neighbourhood of mapping singularities, the robot accuracy will also tend to decrease. As a result, this mapping should be considered in the accuracy computation and in the selection of the legs to observe.

## 4.3.2 Leg Selection Algorithms

### 4.3.2.1 Offline leg selection

Then, if accuracy is needed, the leg selection must guarantee the best final accuracy. To achieve this goal, the following procedure can be used, illustrated on the GS platform:

1. knowing the six leg orientations at the initial and final GS platform configurations, compute the solutions of the forward geometric model of the twenty 3-UPS robots,
2. find, using a procedure similar to the one proposed in (Bonev et al., 2006) for all virtual 3-UPS robots, 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; if it does not exist any 3-UPS robot for which initial and final configurations belong to the same assembly mode, the displacement is not feasible, except if special trajectories are planned as mentioned previously,
3. for all remaining virtual 3-UPS robots, knowing the observation error  $\delta u$ , compute the positioning error using (3.2); retain the set of legs that guarantee the best accuracy;
4. test the controller (in simulation) with the retained set of legs; if there is no problem of convergence and that the legs are observable during the whole displacement, the problem is solved; if not, discard this set of leg and redo point 3; if it does not exist any 3-UPS robot for which initial and final configurations belong to the same assembly mode, the displacement is not feasible, except if special trajectories are planned as mentioned previously.

Obviously, this methodology can be extended when four or five legs are observed. One should also be aware that instead of giving the initial and final robot configurations to the controller, it is better to define a trajectory between these two points in order to avoid crossing singularities inadvertently.

An example of this method is illustrated in Figures 4.6 and 4.5, which represent the positioning error of the GS Platform from Section 3.1. Both times the robot performs a motion from its home position ( $\{x = 0, y = 0, z = 0.3\}$ ) to the desired point at  $\{x = 0, y = -0.06, z = 0.4\}$ . These points were selected based on the accuracy analysis performed in Section 3.1, which can be seen in Figure 3.3. In the first case, legs  $\{1, 3, 5\}$  have been selected, without the use of the algorithm. In the second case, after running the algorithm, it was decided to observe legs  $\{1, 2, 5\}$ . Figures 4.6 and 4.5 show the merit of the algorithm, as well as provide consistent results with Section 3.1

### 4.3.2.2 Online minimal conditioning number leg selection

Another way to solve the problem of crossing singularities is the online selection of legs. If the robot approaches a singularity which is only present when observing a certain set of legs, the controller could choose to observe a different set of legs, whose observation does not include the same singularity, allowing for the motion of the robot to continue. Since approaching a

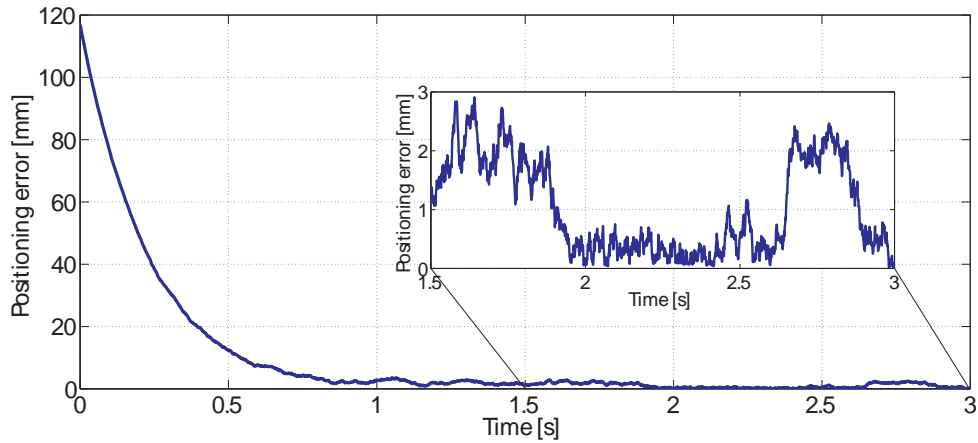


Figure 4.5 – Motion of the GS Platform using legs 1,3,5

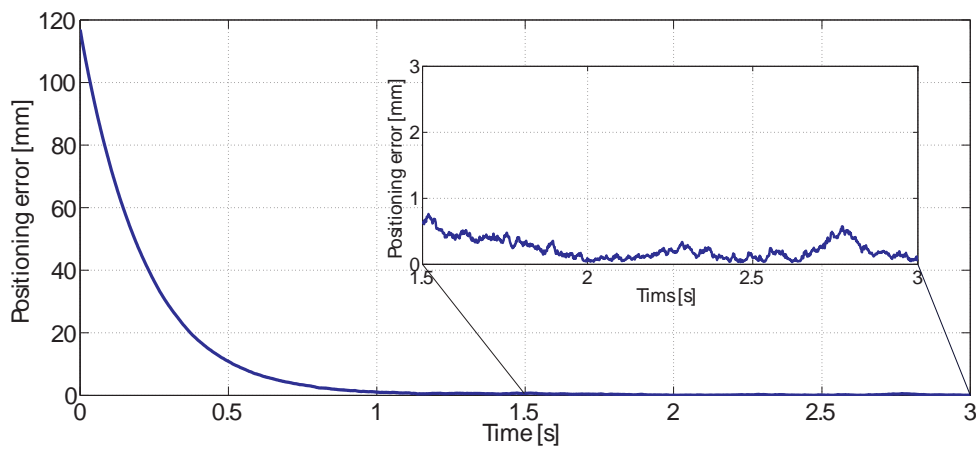


Figure 4.6 – Motion of the GS Platform using legs 1,2,5 (after leg selection algorithm)

singularity is inherently tied to the conditioning number of the interaction matrix, choosing to observe the set of legs whose corresponding interaction matrix has the lowest conditioning number of all possible such matrices should prove to be an optimal leg selection. An algorithm would take the following form:

1. calculate the interaction matrix  $\mathbf{M}_i$  corresponding to each leg at each time instant,
2. stack the interaction matrices forming  $\mathbf{M}_j$  stacked matrices corresponding to  $j = 1, \dots, n$  sets of observed legs,
3. calculate the conditioning number of each stacked matrix  $c_j = \text{cond}(\mathbf{M}_j)$ ,
4. select to observe the set of legs  $s$ , whose interaction matrix  $\mathbf{M}_s$  has the lowest conditioning number  $c_s = \min(c_j)$ ,  $j = 1, \dots, n$ ,
5. if the observed set of legs has changed, perform a smooth change between the two sets of legs using a sigmoid function:

$$\dot{\mathbf{x}} = \sigma \dot{\mathbf{x}}_1 + (1 - \sigma) \dot{\mathbf{x}}_2, \quad (4.5)$$

where  $\dot{\mathbf{x}}_i = \mathbf{M}_i \dot{\mathbf{u}}_i$ , the index  $i = 1$  corresponding to the new set of legs and  $i = 2$  corresponding to the old set of legs, and  $\sigma$  is a function of  $\mathbf{M}_1$  and  $\mathbf{M}_2$ .

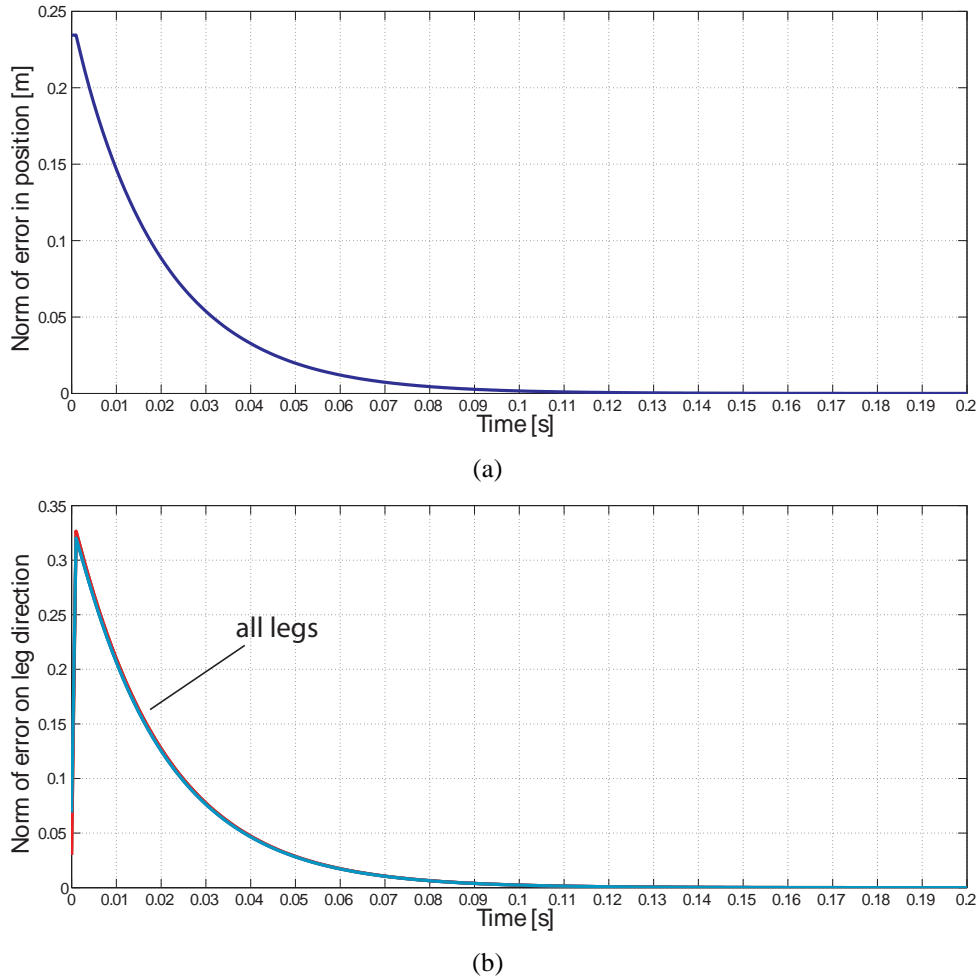


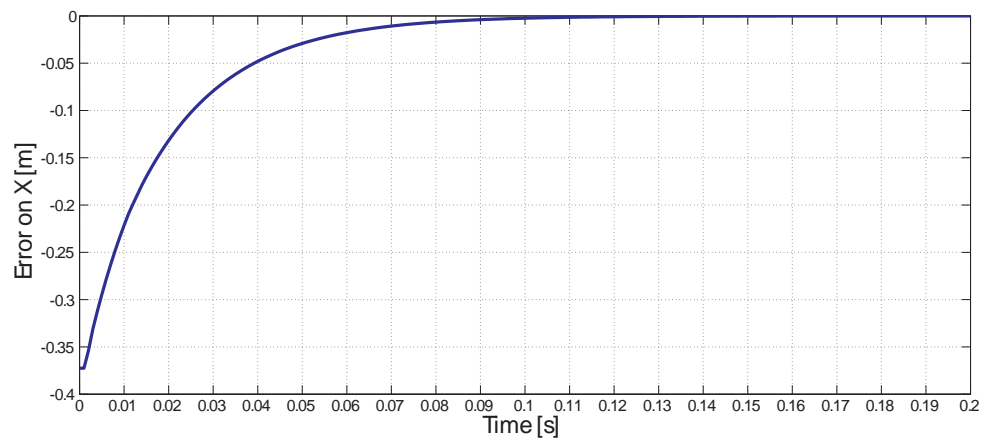
Figure 4.7 – Errors in position and leg directions while using online leg selection

### 4.3.3 Application of the Leg Selection Algorithm to the Adept Quattro

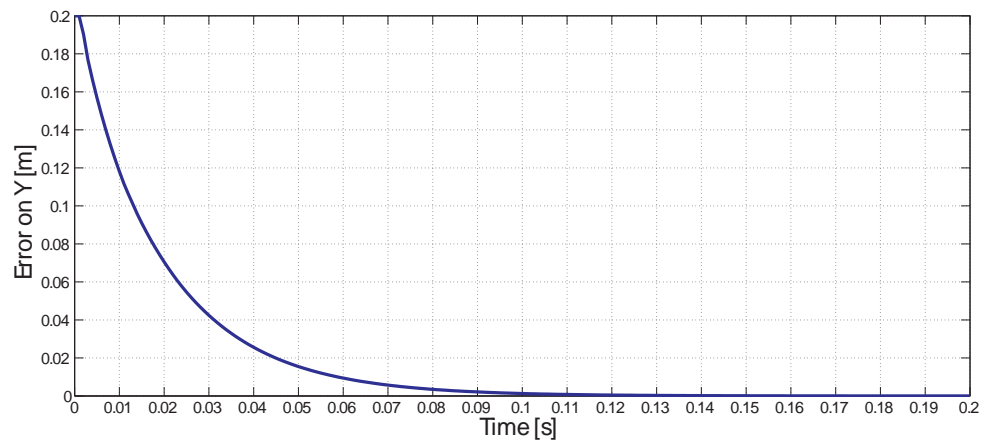
Using the online minimal conditioning number algorithm presented above, the Adept Quattro described in Section 3.3 was made to perform certain movements. First of all, one of the motions presented in Section 3.3.2 was performed, namely the motion starting from point  $\{x = 0m, y = 0m, z = -0.75m, \phi = 0deg\}$  and having as destination the point at  $\{x = -0.2m, y = 0m, z = -0.56m, \phi = 0deg\}$ . The behaviour of the robot in this case was as expected, the algorithm selecting the set of  $\{2, 3\}$  legs to be observed, which resulted in the convergence of all legs to their desired directions (Figure ??), as well as convergence of the platform to the destination point (Figure 4.7(a)).

However, when moving from the same starting point  $\{x = 0m, y = 0m, z = -0.75m, \phi = 0deg\}$  to the destination point  $\{x = 0.4m, y = -0.2m, z = -0.83m, \phi = 0deg\}$ , a change of leg observation occurs. The simulation indicates that while the motion starts with the observation of legs  $\{2, 3\}$ , as this seems to be the optimal set of legs for the initial point, it quickly changes to  $\{1, 4\}$ , finally changing to  $\{2, 4\}$ , which is the optimal set for the destination point.

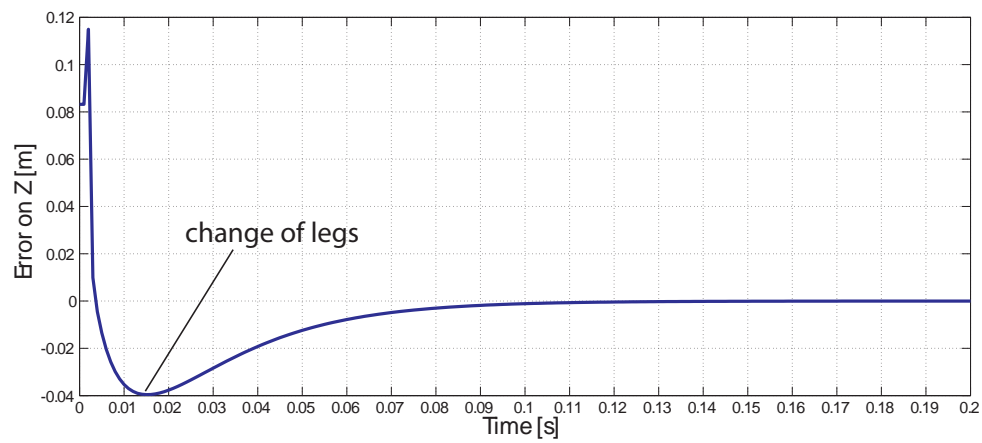
In Figure 4.8, the errors in  $x$ ,  $y$ ,  $z$  and  $\phi$  are presented. The change of legs can be observed in Figures 4.8(c) and 4.8(d), at time step  $t_c = 0.015s$ , when these errors start converging towards zero.



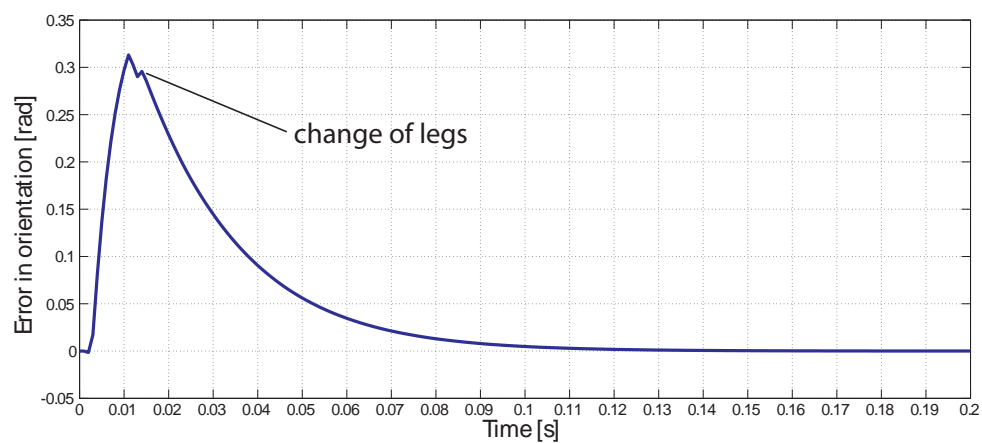
(a)



(b)



(c)



(d)

Figure 4.8 – Errors in position and orientation while using online leg selection

## 4.4 Controllability Analysis Using the Hidden Robot

Thanks to the hidden robot concept, it is possible to analyse the controllability of parallel robots and to define three categories of robots:

1. robots which are not controllable using the leg direction observation: this case will appear if, for a given set of observed features  $s$ , the mapping involved in the controller for estimating the end-effector pose is singular for an infinity of robot configurations (in other words, the end-effector configuration is not observable),
2. robots which are partially controllable in their whole workspace using the leg direction observation: this case will appear if, for a given set of observed features  $s$ , the mapping involved in the controller is not a global diffeomorphism (i.e. a given set of observed features  $s$  may lead to several possible end-effector configurations – Figure ??),
3. robots which are fully controllable in their whole workspace using the leg direction observation: this case will appear if, for a given set of observed features  $s$ , the mapping involved in the controller is a global diffeomorphism (i.e. a given set of observed features  $s$  leads to a unique end-effector configuration – Figure ??).

Families of robots belonging to these categories are defined thereafter. Moreover, after this classification, insights to ensure that all robots could be controllable by adding supplementary observations are provided.

### 4.4.1 Robots Which Are Not Controllable Using the Leg Direction Observation

With the hidden robot concept it is possible to find classes of robots which are not controllable using leg observations, and this without any mathematical derivations. These robots are those with a hidden robot model which is architecturally singular (whatever the number of observed legs). In other words, the hidden robots have unconstrained *dof*.

Three main classes of parallel robots belong to this category (the list is not exhaustive, but groups the most usual and known robots in the community):

- robots with legs whose directions are constant for all robot configurations: these are the cases of planar 3-PPR and 3-PPR robots (Merlet, 2006b; Bonev, 2002) and of certain spatial robots such as the 3-[PP]PS robots<sup>1</sup> (with 6 *dof* – e.g. the MePaM (Caro et al., 2010b)) or 3-PPS robots (with 3 *dof* (Bonev, 2008)). It is obvious that for robots with legs whose directions are constant in the whole workspace, it is not possible to estimate the platform pose from the leg directions only.
- robots with legs whose directions are constant for an infinity of (but not all) robot configurations: this are the cases of PRRRP robots with all P parallel (Figure 4.9(a)) and of Delta-like robots actuated via P joints for which all P are parallel (such as the UraneSX or the I4L (Krut et al., 2003; Company and Pierrot, 2002)). It was shown in (Andreff and

---

1. [PP] means an active planar chain able to achieve two *dof* of translation, such as PP or RR chains

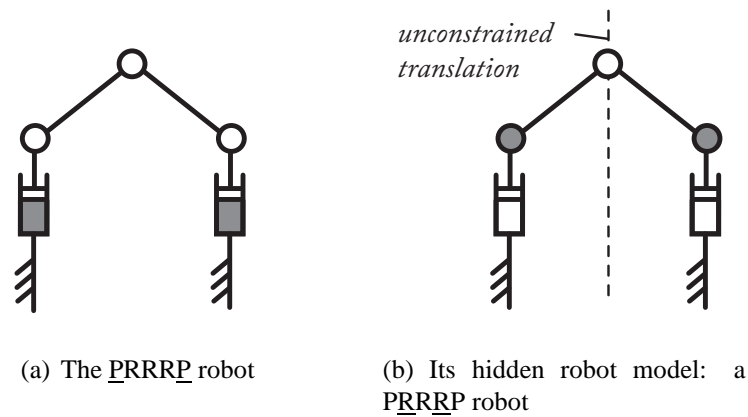


Figure 4.9 – The  $\underline{P}$  $\underline{R}$  $\underline{R}$  $\underline{R}$  $\underline{P}$  robot and its hidden robot model

(Martinet, 2006) through the analysis of the rank deficiency of the interaction matrix that it was not possible to control such types of robots using leg direction observation. Considering this problem with the hidden robot concept is very easy. For example, in the case of the  $\underline{P}$  $\underline{R}$  $\underline{R}$  $\underline{R}$  $\underline{P}$  robot with parallel  $\underline{P}$  joints, the hidden robot has a  $\underline{P}$  $\underline{R}$  $\underline{R}$  $\underline{R}$  $\underline{P}$  architecture (Figure 4.9(b)), where the parallel  $\underline{P}$  joints are passive. This robot is well-known to be architecturally singular as there is no way to control the translation along the axis of the parallel  $\underline{P}$  joints. This result can be easily extended to the cases of the hidden robots of the UraneSX and the I4L.

- robots with legs whose directions vary with the robot configurations but for which all hidden robot legs contain active  $\underline{R}$  joints but only passive  $\underline{P}$  joints: the most known robot of this category will be the planar 3- $\underline{P}$  $\underline{R}$  $\underline{P}$  robot for which the hidden robot model is a 3- $\underline{P}$  $\underline{R}$  $\underline{P}$  which is known to be uncontrollable (Merlet, 2006b; Bonev, 2002).

#### 4.4.2 Robots Which Are Partially Controllable in Their Whole Workspace Using the Leg Direction Observation

The hidden robot model can be used to analyse and understand the singularities of the mapping and to study if a global diffeomorphism exists between the space of the observed element and the Cartesian space. However, not finding a global diffeomorphism does not necessarily mean that the robot is not controllable. This only means that the robot will not be able to access certain zones of its workspace (the zones corresponding to the assembly modes of the hidden robot model which are not contained in the same aspect as the one of the robot initial configuration). This is of course a problem if the operational workspace of the real robot is fully or partially included in these zones.

Robots belonging to this category are probably the most numerous. They are those for which the hidden robot models have several possible assembly modes, whatever is the number of observed leg directions. Presenting an exhaustive list of robots of this category is totally impossible because it requires the analysis of the assembly modes of all hidden robot models for each robot architecture. However, some examples can be provided.



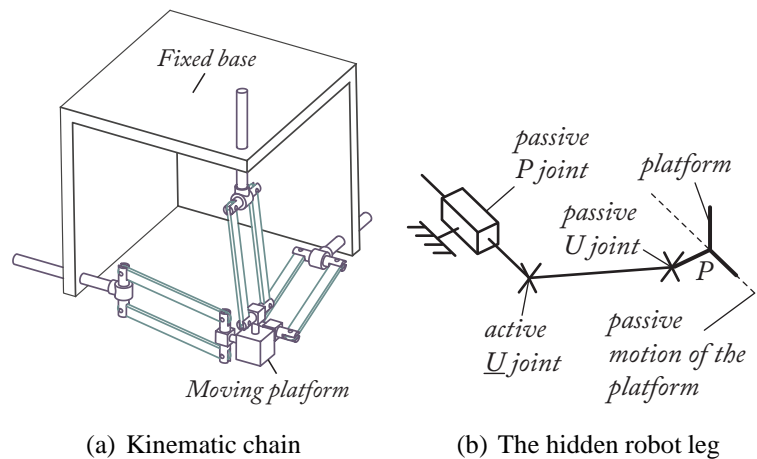


Figure 4.10 – The Orthoglide and its hidden robot leg.

In Section 3.3 it has been shown, both numerically and experimentally, that the Adept Quattro (Nabat et al., 2005) controlled by the leg direction observation has always at least two assembly modes of the hidden robot model, whatever the number of observed legs. As a result, some areas of the robot workspace were never reachable.

It should be mentioned that, even if it is out of the scope of the present paper, it can be verified if the operational workspace of the real robot is fully or partially included in the aspects of the hidden robot models. This problem may be complex, but can be solved using some advanced tools such as interval analysis (Merlet, 2006b) or Cylindrical Algebraic Decomposition (CAD) (Chablat et al., 2011). It should also be mentioned that a *Maple* library for the CAD has been developed by IRCCyN and is available under request on (SiRoPa Toolbox, 2015).

#### 4.4.3 Robots Which Are Fully Controllable in Their Whole Workspace Using the Leg Direction Observation

Robots of this category are those for which a global diffeomorphism between the leg direction space and Cartesian space exists for all workspace configurations. Their hidden robot models have only one possible assembly mode. Once again, presenting an exhaustive list of robots of this category is totally impossible because it requires the analysis of the assembly modes of all hidden robot models for each robot architecture.

However, we show here for the first time robots belonging to this category. Let us consider the Orthoglide (Chablat and Wenger, 2003) designed at IRCCyN (Figure 4.10(a)). This robot is a mechanism with 3 translational *dof* of the platform. It is composed of three identical legs made of either a  $\underline{\text{PRIIR}}$  architecture, or a  $\underline{\text{PUU}}$  architecture, the  $\underline{\text{P}}$  joint of each leg being orthogonal.

Let us consider the second type of legs which is simpler to analyse (even if the following results are also true for the first type of leg). If the link between the two passive U joints is observed, from Section 3.2.2, the hidden robot leg has a  $\underline{\text{PUU}}$  architecture with, of course, two degrees of actuation. As a result, for controlling the three *dof* of the platform, only two legs need to be observed and it could be proven that estimating the robot pose is equivalent to finding the intersection of two lines in space (three lines if all three legs are observed – each line passes



through the corresponding leg tip and is directed along the P joint direction: it corresponds to the free motion of the platform due to the virtual passive P joint of each leg (Figure 4.10(b)). As a result, in a general manner, the  $fkp$  may have:

- zero solution (impossible in reality due to the robot geometric constraints),
- infinite solutions *if and only if* the P joints are parallel (not possible for the Orthoglide as all P joints are orthogonal),
- one solution (the only possibility).

Moreover, a simple singularity analysis of all the possible hidden robot models of the Orthoglide could show that they have no Type 2 singularities (which is coherent with the fact that the  $fkp$  has only one solution).

By extension of these results, it could be straightforwardly proven that all robots with 3 translational *dof* of the platform, or with Schönflies motions (3 translational *dof* of the platform plus one rotational *dof* about one fixed axis), which are composed of identical legs made of either a PRIIR architecture, or a PUU architecture, and have at least two P joints that are not parallel (e.g. the Y-STAR (Hervé, 1992)), are fully controllable in their whole workspace using leg direction observation.

## 4.5 Proper Feature Observation

When presented with the controllability analysis detailed earlier, we can observe certain apparent shortcomings associated with using this controller. However, as we chose the leg directions as the features to observe due to certain criteria mentioned in Section 2.3.1, one may wonder: can we choose different or additional features to observe to overcome the controllability issues raised earlier?

Since we would still be dealing with a mapping between the Cartesian space and the selected feature space, all characteristics pertaining to the hidden robot described throughout this Section and the previous one would still apply, the only difference being the architecture of the hidden robot used.

Without detailing the rules for the construction of these new hidden robots, let us present in what follows an example which illustrates the controllability categories described earlier, while also proving that a robot which normally is uncontrollable using leg direction-based observation can become partially or even fully controllable when additional information is used.

### 4.5.1 Illustrative Example

#### 4.5.1.1 Presentation of the robot under study

In the present section, we illustrate our previous points by analysing the controllability of a special type of 3-PRR robot with parallel P joints and two coincident platform joints (Figure 4.11(a)). In the following, we consider that:

- $q_1, q_2$  and  $q_3$  are the coordinates of the actuators of the real robot,
- the lengths of segments  $A_1P$ ,  $A_2P$  and  $A_3P$  are denoted  $l_{A_1P}$ ,  $l_{A_2P}$  and  $l_{A_3B}$ , respectively, and are equal, i.e.  $l = l_{A_1P} = l_{A_2P} = l_{A_3B}$ ,
- the controlled point on the effector is the point  $P$  with coordinates  $x$  and  $y$  along the  $x$  and  $y$  axes, respectively,
- the orientation of the platform with respect to the  $x$  axis is parametrised by the angle  $\phi$ ,
- the distance between the joints located at points  $P$  and  $B$  is denoted as  $d$ .

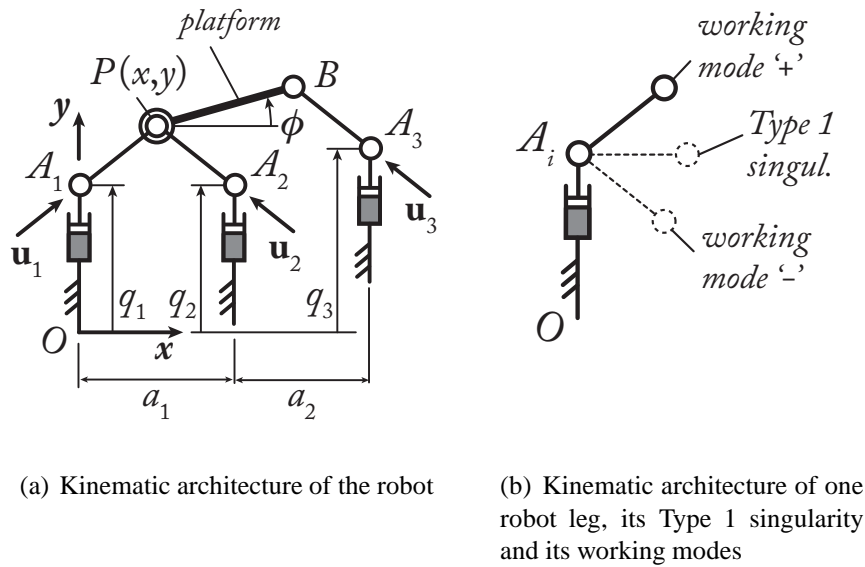


Figure 4.11 – Schematics of the 3-PRR robot.

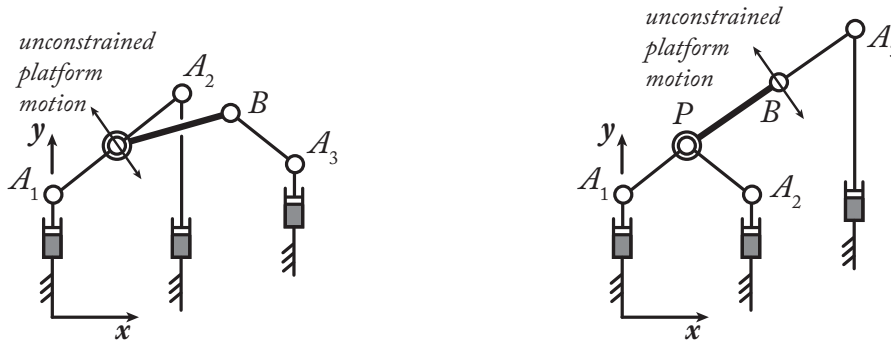
For this mechanism, Type 1 singularities appear when  $\underline{u}_i$  is orthogonal to the direction of the prismatic guide of the leg  $i$  (Figure 4.11(b)). These singularities represent some workspace boundaries.

For this mechanism, Type 2 singularities appear:

- when  $\underline{u}_1$  and  $\underline{u}_2$  are collinear (Figure 4.12(a)): they appear *if and only if* the legs 1 and 2 are in antagonistic working modes ('+-' or '-+', see Figure 4.11(b)) for  $x = a_1/2$  for any  $y$  and  $\phi$ , i.e. they never appear when the legs 1 and 2 are in working modes '++' or '--' such as in Figure 4.11(a).
- or when  $\underline{u}_2$  and  $\overrightarrow{PB}$  are collinear (Figure 4.12(b)): they may appear for any  $x$  and  $y$  *if and only if* the robot reaches constant platform orientations defined by  $\cos \phi = a_2/(d + l)$  or  $\cos \phi = a_2/|d - l|$ .

#### 4.5.1.2 Analysis of the possible hidden robot models

**Case 1:** Let us now assume that we want to control the 3-PRR robot depicted at Figure 4.11(a) by using the observation of its leg directions  $\underline{u}_i$  (see Section ??). From Section 3.2, we know



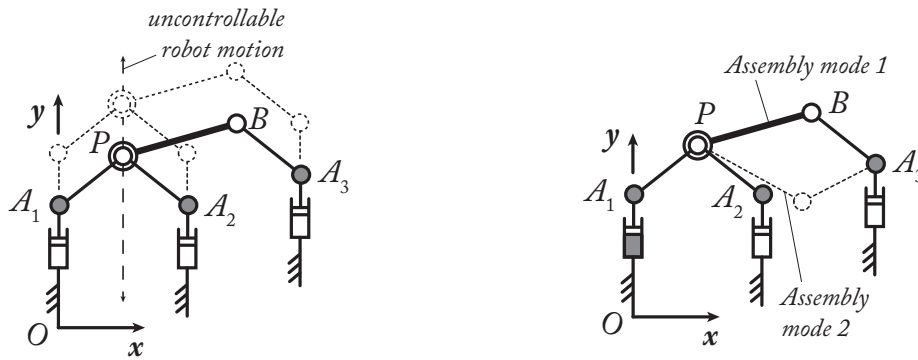
(a) Example of the first case of Type 2 singularity      (b) Example of the second case of Type 2 singularity

Figure 4.12 – Singularities of the 3-PRR robot.

that using such a control approach involves the appearance of a hidden robot model. This hidden robot model can be found by straightforwardly using the results of Section 3.2 and is a 3-PRR robot shown in Figure 4.13(a). This robot is known to be architecturally singular (it can freely move along the  $y$  axis) and can not be controlled by using only the observation of its leg directions  $\underline{u}_i$ .

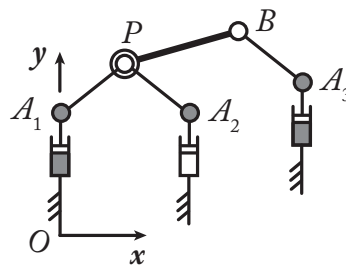
**Case 2:** As a result, one would logically wonder what should be the necessary information to retain in the controller to servo the robot. By using the results of Section ??, we know that, from the projection of the cylindrical leg in the image plane, it is not only possible to estimate the leg direction, but also the Plücker coordinates of the line passing through the axis of the cylinder, i.e. the direction and location in space of this line. Let us consider that we add this information for the estimation of the leg 1 position only. Modifying the hidden robot model according to Figure 4.14(a), the corresponding robot model hidden in the controller is depicted in Figure 4.13(b): this is a PRR-{2-PRR} robot which is not architecturally singular. In other words, using the Plücker coordinates of the line for leg 1 involve to actuate both the first P and R joints of the corresponding leg, i.e. the virtual leg is a PRR leg. For the PRR-{2-PRR} robot, it is possible to prove that two assembly modes exist which are separated by a Type 2 singularity at  $\phi = 0$  or  $\phi$  (for any  $x$  and  $y$ ). For both assembly modes, the end-effector position is the same, while the orientation is different. Thus, the robot is not fully controllable in its whole workspace.

**Case 3:** From the result that, using the Plücker coordinates of the line passing through the axis of the cylinder, the leg of the virtual robot becomes a PRR leg, it is possible to understand what is the minimal set of information to provide to the controller to fully control the robot in the whole workspace: we need to use the Plücker coordinates of the lines passing through legs 1 and 3 and the direction of the leg 2. In such a case, the hidden robot model is a PRR-{2-PRR} robot depicted in Figure 4.13(c). It is possible to prove that this robot has no Type 2 singularity and can freely access its entire workspace.



(a) When all leg directions  $\underline{u}_i$  are observed (Case 1): a 3-PRR robot

(b) Case 1: When all leg directions  $\underline{u}_i$  and the Plücker coordinates of the line passing through the leg 1 are observed (Case 2): a PRR-{2-PRR} robot



(c) When all leg directions  $\underline{u}_i$  and the Plücker coordinates of the lines passing through the legs 1 and 3 are observed (Case 3): a PRR-{2-PRR} robot

Figure 4.13 – Hidden robots involved in the tested visual servoings of the 3-PRR robot.

#### 4.5.1.3 Simulation results

Simulations are performed on an Adams mockup of the 3-PRR robot with the following values for the geometric parameters:  $l = 1$  m,  $d = 0.4$  m,  $a_1 = 0.4$  m and  $a_2 = 0.25$  m. This virtual mockup is connected to Matlab/Simulink via the module Adams/Controls. The controller presented in Section 2.3.2 is applied with a value of  $\lambda$  assigned to 20.

The initial configuration of the robot end-effector is  $x_0 = 0.20$  m,  $y_0 = 0.98$  m and  $\phi_0 = -45$  deg. We want to reach the end-effector configuration  $x_f = 0.20$  m,  $y_f = 1.03$  m and  $\phi_f = -10$  deg. For that, we use the three possible controllers (Cases 1, 2 and 3) proposed in the previous Section and simulate the robot behaviour with the Adams mockup during 1 second. For the three cases, the errors on the used observed features (either the leg directions or the Plücker coordinates of the lines) tends to zero at the end of the simulation. However, this is not necessarily the case for the end-effector configuration (Table 4.2).

With the controller of Case 1 based on the observation of the leg directions only, the robot is not able to attain the final end-effector configuration. Moreover, the end-effector position is unchanged (while its orientation has been modified) which is coherent with the results of the previous section: the corresponding hidden robot is architecturally singular and its motion along the  $y$  axis is uncontrollable.

Table 4.2 – Final end-effector configuration for the desired end-effector configuration  $x_f = 0.20$  m,  $y_f = 1.03$  m and  $\phi_f = -10$  deg

	$x$ (m)	$y$ (m)	$\phi$ (deg)
Case 1	0.20	0.98	-10
Case 2	0.20	1.03	-10
Case 3	0.20	1.03	-10

For the two other controllers, the convergence towards the desired end-effector pose is achieved.

Now, we change the desired end-effector configuration  $x_f = 0.20$  m,  $y_f = 1.03$  m and  $\phi_f = +10$  deg. The results for the end-effector convergence are provided in Table 4.3.

Table 4.3 – Final end-effector configuration for the desired end-effector configuration  $x_f = 0.20$  m,  $y_f = 1.03$  m and  $\phi_f = +10$  deg

	$x$ (m)	$y$ (m)	$\phi$ (deg)
Case 1	0.20	0.98	-10
Case 2	0.20	1.03	-10
Case 3	0.20	1.03	+10

With the controller of Case 1, the results are unchanged: the robot is not able to reach the desired configuration.

With the controller of Case 2 based on the observation of the Plücker coordinates of the line passing through the leg 1 and the other leg directions, the robot attains the final end-effector position, but not the correct orientation. This is coherent with the results of the previous section: the corresponding hidden robot has two assembly modes with similar end-effector positions but different orientations. It can be proven that, for the given robot geometric parameters, the two assembly modes of the  $\text{PRR}\{-2\text{-PRR}\}$  robot for the given observed features at the desired final robot configuration are:

- $x_1 = 0.20$  m,  $y_1 = 1.03$  m and  $\phi_1 = +10$  deg, and
- $x_2 = 0.20$  m,  $y_2 = 1.03$  m and  $\phi_2 = -10$  deg.

Thus, the robot has converged towards the second assembly mode, which was not the desired one. However, this second assembly mode was reached during the first simulation, because it is enclosed in the same workspace aspect corresponding to the initial robot configuration.

Finally, with the controller of Case 3 based on the observation of the Plücker coordinates of the lines passing through the legs 1 and 3 and the leg 2 direction, the robot reach the desired configuration.

#### 4.5.1.4 Discussion

To conclude this part, it is necessary to mention that:

- in our simulations, we have considered that the observed features were not noisy, which is not true in reality. This has been simply assumed for two main reasons: (i) robustness of these types of controllers has already been shown in previous works (e.g. (Andreff et al., 2005, 2007; Rosenzveig et al., 2014)) and (ii) adding noise would have made the analysis of the convergence results in the controllers of Case 1 and 2 more difficult to explain, without bringing any added value to these simulations.
- the results for the controller of Case 3 would have been the same if the Plücker coordinates of the line 2 were observed instead of those of the line 1. The choice of the best leg to observe could have been done by a procedure presented in (Briot and Martinet, 2013) which ensure to select the legs that lead to the best end-effector accuracy. However, this was out of the scope of the present paper.
- in the whole paper, it is considered that the sensor measurement space is the same as the leg direction space. However, for example using a camera, the leg directions are not directly measured but rebuilt from the observation of the legs limbs projection in the 2D camera space (Andreff et al., 2005). Thus, for the leg reconstruction, the mapping between the camera space and the real 3D space is involved, and it is not free of singularities (see (Michel and Rives, 1993) for an example of mapping singularities). In the neighbourhood of mapping singularities, the robot accuracy will also tend to decrease. As a result, this mapping should be considered in the accuracy computation and in the selection of the legs to observe.

### 4.5.2 Robots Which Become Fully Controllable in Their Whole Workspace if Additional Information Is Used

Using the example presented above, we can append the categories outlined in Section 4.4 with a new category, namely robots which can become fully controllable by adding additional information in the controller.

For example, it has been very recently proven in (Vignolo et al., 2014) that, from the projection of the cylindrical leg in the image plane (Figure 2.13), it is not only possible to estimate the leg direction, but also the Plücker coordinates of the line passing through the axis of the cylinder, i.e. the direction and location in space of this line. Using this information leads to a modification of the virtual leg as shown in Figure 4.14(a): the additional prismatic chain, instead of being passive, becomes active.

This additional information can solve many issues of controllability mentioned above. For example, by estimating the Plücker coordinates of the line passing through its legs, the  $\underline{\text{PRRRP}}$  robot of Section 4.4.1 becomes controllable as the hidden robot model becomes a  $\underline{\text{PRRRP}}$  robot (Figure 4.14(b)), which is fully controllable.

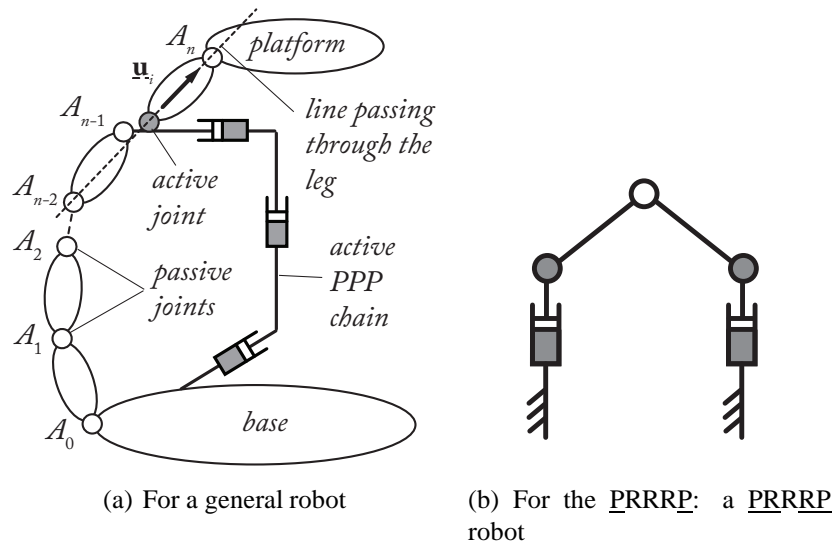


Figure 4.14 – The hidden robot leg when the Plücker coordinates of the line passing through the axis of the leg are observed.

However, this information may not be enough for some categories of robots, such as for the MePaM (Caro et al., 2010b) for which it has been shown in (Vignolo, 2014) that using the Plücker coordinates of the line passing through the legs leads to a robot which is partially controllable in its whole workspace (eight different assembly modes of the hidden robot model may appear). A similar result could be proven for the GS platform for which the Plücker coordinates do not bring any additional useful information in the controller. For such robots, two main solutions are possible:

- if the robot operational workspace is included in one given aspect of the hidden robot model, the controller may be sufficient to fully control the robot in its operational workspace,
- other features (such as other robot elements (joint locations, other links, etc.)) should be observed to complete the missing information.

## 4.6 Conclusions

This chapter presents additional features of the hidden robot concept, proving its usefulness by providing answers to complex questions through simple applications.

The hidden robot concept was applied to families of planar and spatial parallel mechanisms, exploring their different assembly modes and mapping singularities through straightforward geometric means, without the need to rely on complicated analyses. The hidden robot model remains valid when applied to these various robot architectures, proving the general nature of the approach.

The importance of leg selection is then presented, along with proposed algorithms which ensure correct and accurate convergence of the platform to the desired end-effector pose. Simulations are performed which illustrate and validate this algorithm.

Robots are then categorised according to their controllability using the hidden robot derived from leg direction-based observation. Then, other feature observation is proposed to tackle the problem of uncontrollability. This is then illustrated with an example, validated through simulations.

The results presented above show the real added value of using the hidden robot concept. The hidden robot being a tangible visualization of the mapping between the observation space and the real robot Cartesian space, it is possible:

- to prove if the studied robot is controllable or not in its whole workspace by the use of quite simple mechanism analysis tools,
- to understand the features to observe to ensure the controllability of the robot in its whole workspace.

In the following chapter, additional use for the hidden robot concept is presented: it will be integrated into the design process with much ease, ensuring the controllability of the final robot using the desired observed features.



# The Hidden Robot as a Sensor-Based Design Tool

*This chapter integrates the hidden robot concept into the design process in an attempt to establish sensor-based design specification. In order to do this, a five-bar robot is geometrically optimised for an externally controlled pick-and-place task, using vision sensors. Two optimisation algorithms are specified, according to the design methodology used. First, a classical design approach governs the optimisation, in which the externally controlled nature of the task is not taken into account. Second, a vision-based design approach is adopted, which uses the hidden robot to model the interaction between the external camera and the scene. The two models are compared proving that the hidden robot must be used to incorporate the vision-based control nature of the task into the design process.*

## 5.1 Classical Design Methodology

### 5.1.1 Classical Design Approach

Because of this, it is important to perform a rigorous study during design in order to produce optimal robot architectures. The design methodology proposed by French ([French and Council, 1985](#)) is separated into four main phases (Figure 5.1): (1) specifications of the product requirements coming from need definition, (2) conceptual design, during which preliminary concepts are proposed and the best designs alternatives are selected, (3) the embodiment of schemes, during which concepts are developed and analysed, and (4) the detailed design that leads to CAD models and manufacturing of prototypes.

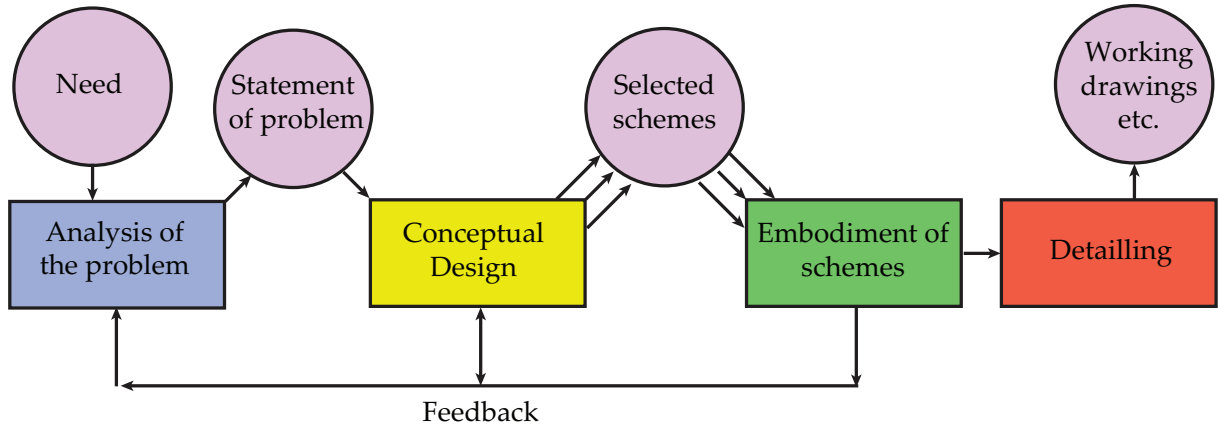


Figure 5.1 – Typical French design methodology

### 5.1.2 Classical Design Optimisation

Traditionally, manipulator design aims at achieving certain geometric goals (i.e. workspace size, dexterity) while optimising geometric parameters with respect to desired kinematic and/or dynamic requirements (i.e. velocity, accuracy, force transmission) (Briot et al., 2010).

To formulate the design problem, let us define the manipulator geometry as the mapping  $g : \Phi \rightarrow \mathbf{W}$ , where  $\Phi$  denotes the configuration space and  $\mathbf{W}$  signifies the workspace. Then, for each point  $P$  within the workspace, we can define a set of matrices  $\mathbf{K}_\alpha(\mathbf{P}, \boldsymbol{\pi})$ , describing various mechanical properties  $\alpha$  of the of the manipulator, given a set of design parameters  $\boldsymbol{\pi}$ . Also, we define physically consistent scalar measures  $\sigma_\beta(\mathbf{K})$  attributed to each of the mechanical properties matrices, which can be used directly in the design as constraints or objectives. Additionally, we introduce indices for the evaluation of global performance of the manipulator  $\eta_\gamma(g, \boldsymbol{\pi})$ , which depend both on the adopted structure and the design parameters.

Finally, we can state the optimisation problem as achieving the best value for the performance indices:

$$\eta_\gamma(g, \boldsymbol{\pi}) \rightarrow \min_{\boldsymbol{\pi}}, \quad \forall \gamma \quad (5.1)$$

subject to the constraints

$$\sigma_\beta(\mathbf{K}_\alpha(\mathbf{P}, \boldsymbol{\pi})) \in S, \quad \forall \alpha, \beta \quad (5.2)$$

that must be satisfied in all points of the desired workspace  $\mathbf{W}_0$ , which includes the desired task.

Since this cannot be solved by direct search methods, a discretisation scheme is utilised, which divides the manipulator workspace  $\mathbf{W}$  into a square matrix, by using different step for each axis based on the shape of the regular desired workspace  $\mathbf{W}_0$ . At each node of the grid the local constraints are evaluated, then the largest sub-matrix of the grid which satisfies all constraints  $\mathbf{W}_L$  is compared to the desired workspace to ensure it is larger.

## 5.2 Optimal Design of the Five-Bar Using the Classical Approach

Let us apply the aforementioned methodology to a robot performing a pick-and-place task. The specifications of the robot, given within the scope of the project, are summed up in Tab. 5.1. The robot should be as compact as possible due to some industrial constraints.

Table 5.1 – Specifications for the robot

Velocity max $v^{lim}$	$6 \text{ m}\cdot\text{s}^{-1}$
Error resolution $e^{max}$	0.5 mm
Cycle time max	200 ms
Path	$25 \text{ mm} \times 300 \text{ mm} \times 25 \text{ mm}$
Regular workspace size $\mathbf{W}_0$	$800 \text{ mm} \times 100 \text{ mm}$
Payload	0.5 Kg

Additionally, a project partner imposes the use of the ETEL TMB140-70 direct drive motors for the actuation, whose characteristics are given in Tab.5.2 as follows:  $V_{max}$  is the maximal motor velocity,  $T_{peak}$  is the peak torque,  $T_C$  is the continuous torque,  $\Phi$  is the motor external diameter,  $J$  is the rotor inertia, and  $res$  is the encoder resolution.

Table 5.2 – Datasheet of the ETEL TMB140-70 motor

$V_{max}$ [rpm]	$res$ [pt/rev]	$T_{peak}$ [Nm]	$T_C$ [Nm]	$\Phi$ [mm]	$J$ [Kg·m <sup>-2</sup> ]
600	200000	89.1	45	166	$2.3e^{-3}$

Additionally, the robot is to be controlled using leg-direction-based visual servoing. The camera designated for this task was a Mikrotron EoSens 4CXP CoaXPress, with the following characteristics (Tab. 5.3):  $R$  is the resolution,  $FR$  is the frame rate at maximum resolution. The lens used with the camera is a Kowa LM12XC, with the following specifications:  $f$  is the focal length,  $I$  is the iris range.

Table 5.3 – Datasheet of the Mikrotron EoSens 4CXP CoaXPress camera with Kowa LM12XC lens

$R$ [px × px]	$FR$ [fps]	$f$ [mm]	$I$ [mm]
$2336 \times 1728$	563	12	$f/2.00 - f/22$

The aforementioned methodology was used in the optimal design of a five-bar mechanism aimed at performing a pick-and-place operation, which would result in a dexterous regular workspace  $\mathbf{W}_0$  of length  $l_{W_0} = 800\text{mm}$  and height  $h_{W_0} = 100\text{mm}$ , in which the specified performance criteria would be met (defined below).

The objective function of the optimisation process was to minimise the planar footprint of the robot  $A = HL$  in its home position (when the angle between the proximal and distal

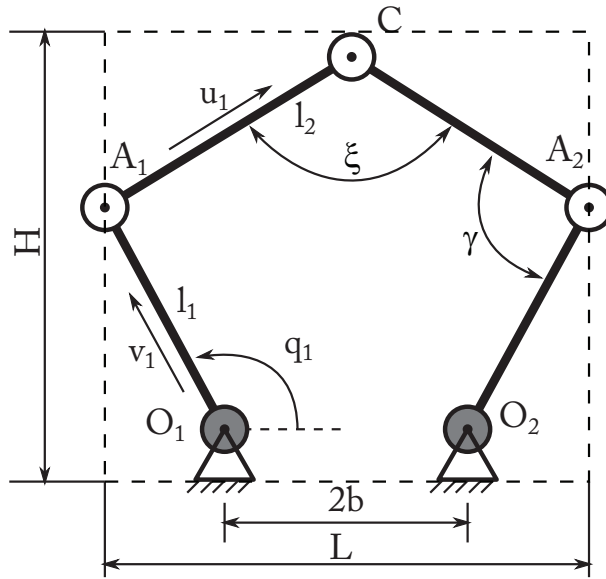


Figure 5.2 – The planar footprint of the five-bar in its home position

links is  $\gamma = \pi/2$ ), thus ensuring better rigidity, and in turn higher stability during operation (Figure 5.2).

The design variables to be optimised were the geometric parameters of the robot: the lengths of the proximal  $l_1$  and distal  $l_2$  links, and semi-distance  $b$  between the anchor points ( $2b = d$ ).

The resulting largest regular dexterous workspace  $\mathbf{W}_L$  of the robot (represented by its length  $l_{W_L}$  and height  $h_{W_L}$ ) should be larger than the required dexterous workspace mentioned earlier.

Consequently, the design optimisation problem is formulated as follows:

$$\begin{aligned}
 &\text{minimise} && A = LH \\
 &\text{over} && \mathbf{x} = [l_1 \ l_2 \ b]^T \\
 &\text{subject to} && l_{W_L} \geq l_{W_0} \\
 &&& h_{W_L} \geq h_{W_0}
 \end{aligned} \tag{5.3}$$

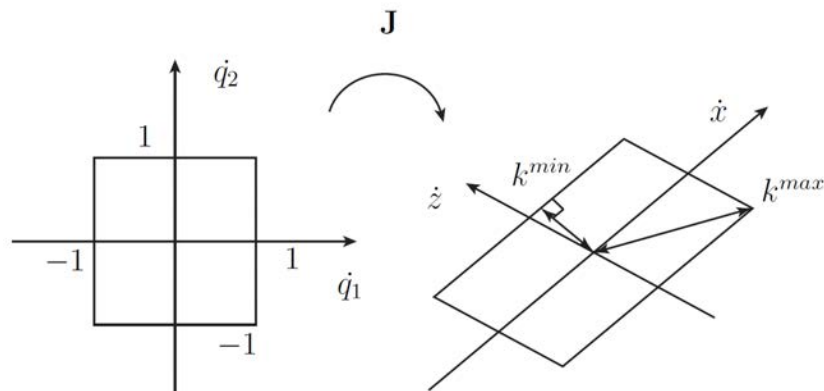


Figure 5.3 – Transmission factor

**Definition:** A Regular Dexterous Workspace is defined such that in each point within it the robot has to meet the following constraints:

1. assembly condition: ensuring that the mechanism can be assembled;
2. singularity-free: ensuring that the robot does not encounter any Type 1 or Type 2 singularities;
3. required velocity transmission: knowing the maximum motor velocity ( $V_{max}$  from Table 5.2) and the kinematic Jacobian  $\mathbf{J}$ , we can obtain the minimum velocity transmission (Figure 5.3, as described in (Briot et al., 2010)):

$$\dot{p}^{min} = k^{min} V_{max} \quad (5.4)$$

which should be greater than the required  $v^{lim}$  (Table 5.1);

4. required transmission angle: the forces exerted into the passive joints are proportional to  $1/\sin \xi$ ,  $\xi$  being the angle between the distal links (Figure 5.2). Consequently, it is decided that  $\sin \xi$  should be higher than 0.1 to avoid excessive efforts in the joints;
5. required error transmission: in a fashion similar to the velocity transmission, knowing the resolution of the motor encoders ( $res$  from Table 5.2), we can obtain the maximum error transmission:

$$\delta p^{max} = k^{min} res/2 \quad (5.5)$$

which should be smaller than the given maximum allowed error  $e^{max}$  (Table 5.1).

To facilitate this optimisation process, the workspace of the robot is divided into a grid, with a different step on the  $x$  and  $y$  axes, such that the ratio between the two corresponds to that of the desired regular dexterous workspace  $\mathbf{W}_0$ . Thus, the matrix representing the grid,  $\Omega$ , is square. In each point of the discretised workspace, the aforementioned constraints are evaluated, and a 1 is placed in the corresponding node of  $\Omega$ , if all the conditions are met, otherwise, we place a 0. Afterwards, we use an algorithm to find the largest square matrix within  $\Omega$  whose elements are 1, like the one used in (Germain et al., 2013). The matrix is then converted back into the Cartesian space through the discretisation steps used before, to obtain the largest regular dexterous workspace,  $\mathbf{W}_L$ .

The results of the optimisation can be seen in Table 5.4.

Table 5.4 – Results of the classical optimisation

$A$ [m <sup>2</sup> ]	0.1144
$l_1$ [m]	0.2219
$l_2$ [m]	0.3863
$b$ [m]	0.1071

When we plot the hidden robot singularities, however, we observe that these cross the largest regular dexterous workspace of the robot (Figure 5.4), due to this not being taken into account during the design. During the simulations we will see how this impacts the performance of the robot.

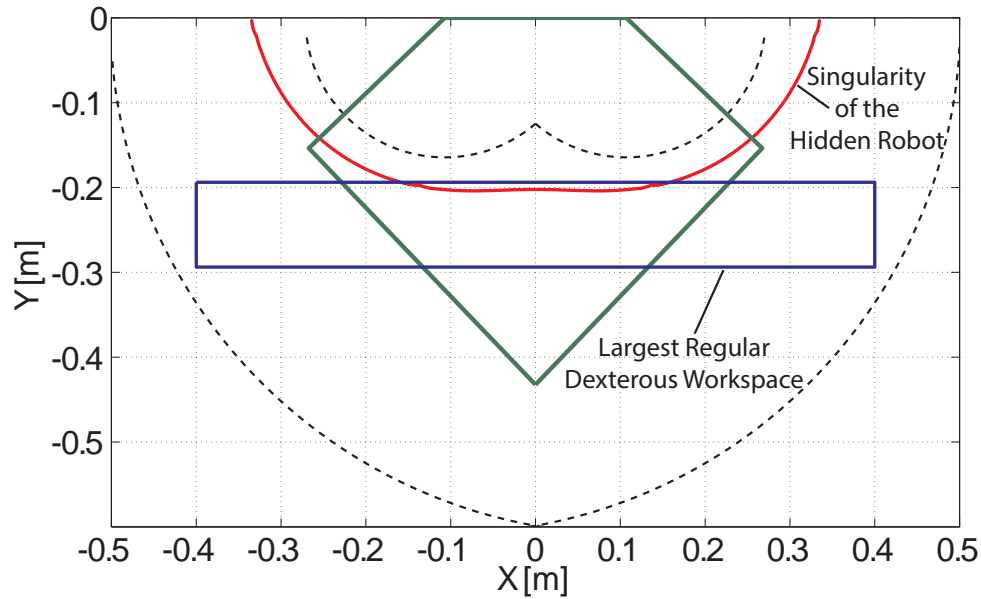


Figure 5.4 – Largest regular dexterous workspace and hidden robot singularities for the camera-optimised architecture

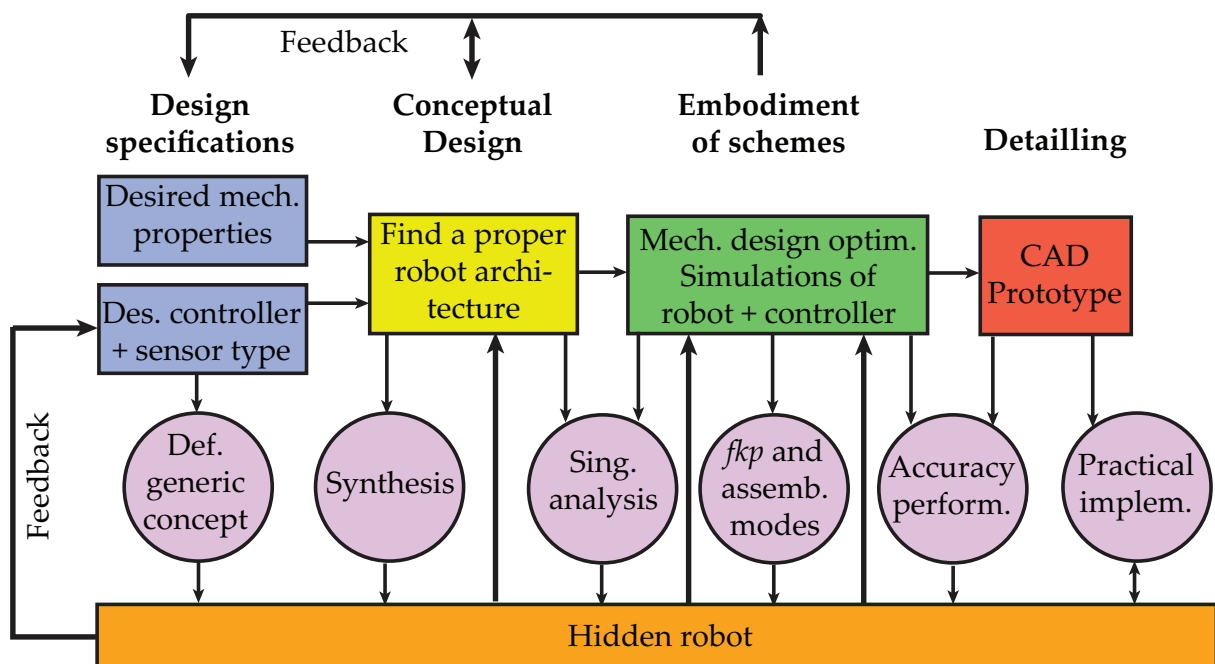


Figure 5.5 – Integration of the hidden robot concept into the design process

### 5.3 Sensor-Based Design Methodology

As mentioned in Chapter 3, if, for a given application, it is specified that a robot must be controlled via the use of exteroceptive sensors, it must be known that such a requirement leads to the definition, for any kind of robot architecture, to a generic hidden robot model.

The hidden robot concept simplifies the analysis of the controller singularities and can be efficiently applied to control design, by using tools and methodologies developed by the mechanical community. We will now discuss how it can be incorporated into the design process.

During the conceptual design phase, the hidden robot of each proposed architecture can be

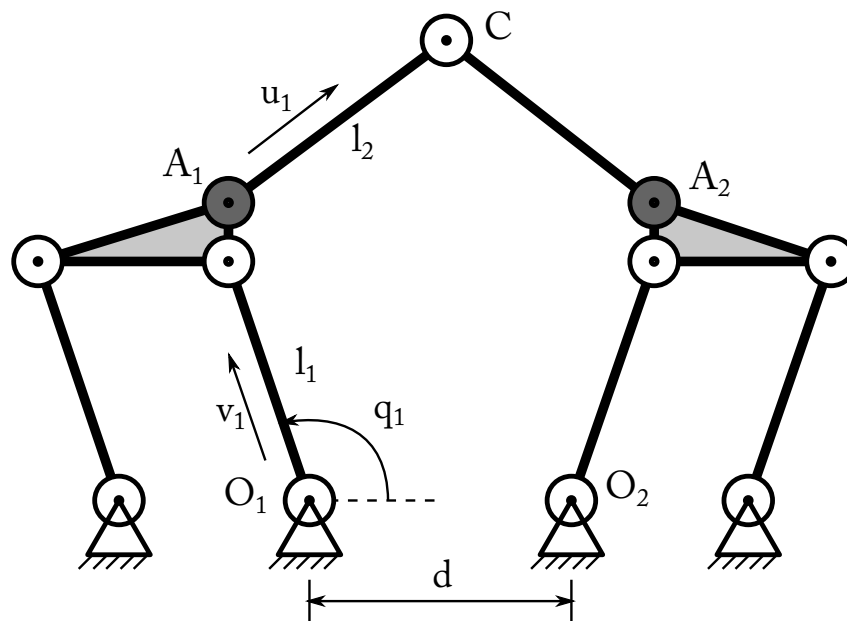


Figure 5.6 – Hidden robot corresponding to the five-bar mechanism

found and compared, giving feedback on which architecture is more suitable for the given needs, e.g. has the least hidden robot singularities in the workspace. If no architecture corresponds to the needs involving their hidden robots, the design specification concerning the sensors used and features observed can be modified. Then, when an appropriate architecture is selected, in the detailing phase the hidden robot can be used to extract performance indices (e.g. wrt accuracy), define a controller and simulate the robot behaviour.

We remind the reader that although this work is limited to using cameras to extract the leg directions of the controlled robot, the hidden robot concept and the methodology described herein are valid for any other type of sensor used and/or feature observed.

## 5.4 Optimal Design of the Five-Bar Using the Sensor-Based Approach

If we take this hidden robot into consideration during the optimisation process, alongside the real robot, we will have accounted for the mapping introduced by our choice of control scheme, and thus avoid the problems encountered during the previous optimisation process. We have chosen to use, in the case of the five-bar mechanism, the hidden robot presented in Figure 5.6.

In addition to the geometric parameters mentioned in Section 5.2, three other design variables are introduced which influence the observation:  $y_c$  and  $z_c$  defining the position of the camera with respect to the world frame, and  $r$  the radius of the distal links to be observed.

Having introduced the additional design variables, the new design problem takes the following form:

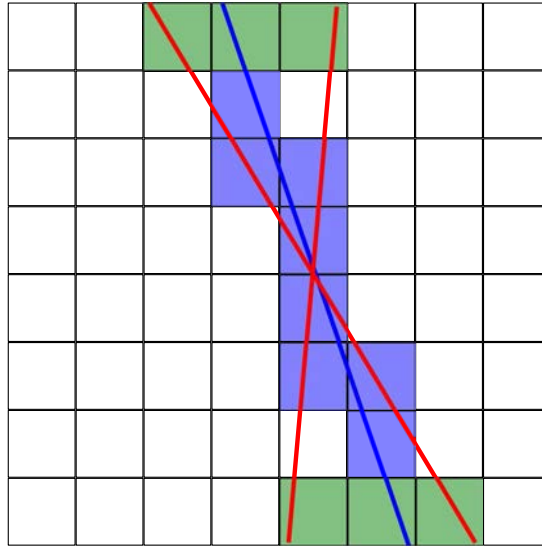


Figure 5.7 – Result of a 1 pixel error on the boundary intersect of the observed line

$$\begin{aligned}
 &\text{minimise} && A = LH \\
 &\text{over} && \mathbf{x} = [l_1 \ l_2 \ b \ y_c \ z_c \ r]^T \\
 &\text{subject to} && l_{W_L} \geq l_{W_0} \\
 &&& h_{W_L} \geq h_{W_0}
 \end{aligned} \tag{5.6}$$

The same discretisation of the workspace is used, however, we have some changes in the local conditions to be evaluated, which determine whether the workspace is dexterous and satisfies the specified needs.

While conditions 1-4 are common and are present within this optimisation process as well, instead of the previous condition 5, three other conditions appear, related to the observation:

- 5b) hidden robot singularity-free: ensuring that the hidden robot does not encounter any Type 2 singularities (corresponding to singularities of the mapping);
- 6b) end-effector in image: ensuring that the end-effector is within the image frame and thus the legs can be observed;
- 7b) required error resolution: due to the fact that the robot is to be controlled using vision, we wish the optimisation to reflect this in terms of error resolution. In a real scenario, errors when using vision appear due to noise in the image. In our case this was reproduced by a random shift in the pixels where the image projection of the leg edges meet the frame boundary. An error of 1 pixel was considered, which is illustrated in Figure 5.7. Like in the previous case, mapping this error into Cartesian space should have an effect on the end-effector lower than the maximum allowed error resolution ( $e^{max} = 0.5\text{mm}$ ).

During the optimisation process we have noticed that the value for the observed leg radius consistently tended for the selected upper bound. To investigate the influence of the leg radius on the error resolution, which is the fundamentally different criterion between the two optimisation processes, we selected fixed values for the other design parameters and varied only the leg



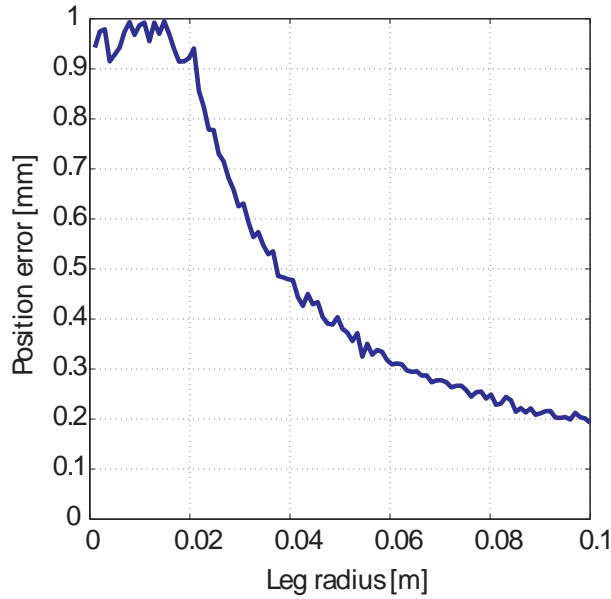


Figure 5.8 – Influence of the leg radius on the end-effector error resolution

radius (values of the fixed parameters:  $l_1 = 0.321\text{m}$ ,  $l_2 = 0.437\text{m}$ ,  $b = 0.083\text{m}$ ,  $y_c = -0.5\text{m}$ ,  $z_c = 0.9\text{m}$ ).

As can be seen in Figure 5.8, the error resolution decreases along with the increase of the leg radius. This is due to the fact that a larger leg radius results in the projection of the leg edges in the image being further apart; this results in a more robust result, since the leg direction is obtained through a cross product of the projected leg edges, (Eq.2.49).

As such, to ensure that we can obtain the desired 0.5mm error resolution, we have fixed the leg radius to be 0.04m on subsequent optimisations.

The results of the optimisation can be seen in Table 5.5.

Table 5.5 – Results of the vision-based optimisation

$A$ [m <sup>2</sup> ]	0.1156
$l_1$ [m]	0.2291
$l_2$ [m]	0.3750
$b$ [m]	0.1092
$r$ [m]	0.0400
$y_c$ [m]	-0.4340
$z_c$ [m]	0.5908

It seems from these results that this second design is less compact than the original one, due to the value of the objective function being slightly higher. However, when we plot the hidden robot singularities within the workspace (Figure 5.9), we can observe that, contrary to the first design, the largest regular dexterous workspace is free of such singularities. Consequently, the error resolution on the positioning of the end-effector due to errors in the image are less than the admissible 0.5mm. This is further reinforced by the simulations in the next section.

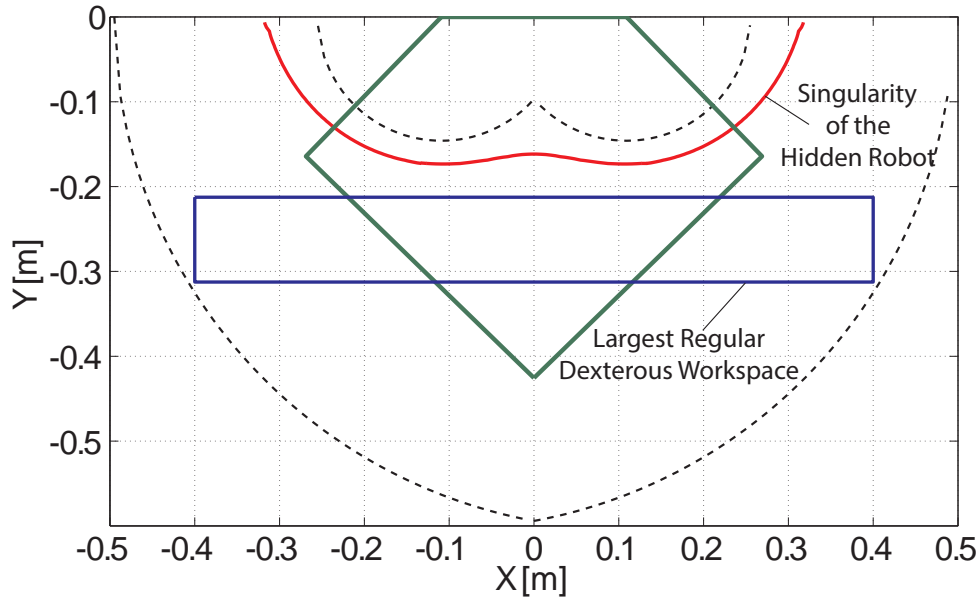


Figure 5.9 – Largest regular dexterous workspace and hidden robot singularities for the camera-optimised architecture

## 5.5 Design Comparison

Simulations were performed in a connected ADAMS-Simulink environment. The manipulator was made to move from its home position (depicted in Figure 5.2) to different positions within the workspace, notable in the uppermost point, represented by  $\{x = 0m, y = -0.2m\}$  in the case of the encoder-optimised design and  $\{x = 0m, y = -0.2125m\}$  in the case of the camera-optimised design. This point is important, because in the case of the optimisation when the hidden robot was not taken into account, it lies on the opposite side of the hidden robot singularity, with respect to the starting point.

When the encoder-optimised robot was made to move to this point using the control law given by 2.58, it was unable to reach the desired position. Indeed, while the observed legs converged to their desired directions (Figure 5.10(a)), there was still an error present on the position (Figure 5.10(b)). This is the same phenomenon as in the case of the Quattro presented in Section 3.3.

In the case of the camera-optimised design, both the leg directions and the position converged to their respective desired values (Figure 5.11). This is due to the hidden robot being taken into account during the geometric parameter optimisation, which allowed the selection of parameters in such a way as to eliminate the hidden robot singularity from the desired workspace.

Additionally, the accuracy of the encoder-optimised design diminishes in the proximity of the singularities. This can be seen in the following set of simulation results. The controller which was responsible for the movement was a leg-direction-based visual servoing controller, was governed by the control law given at 2.58. To simulate noise, an error was applied to the observation, represented by a random shift of 1 pixel in the projection of the leg edges from which the leg direction is calculated, and then the end-effector pose reconstructed. Then, we

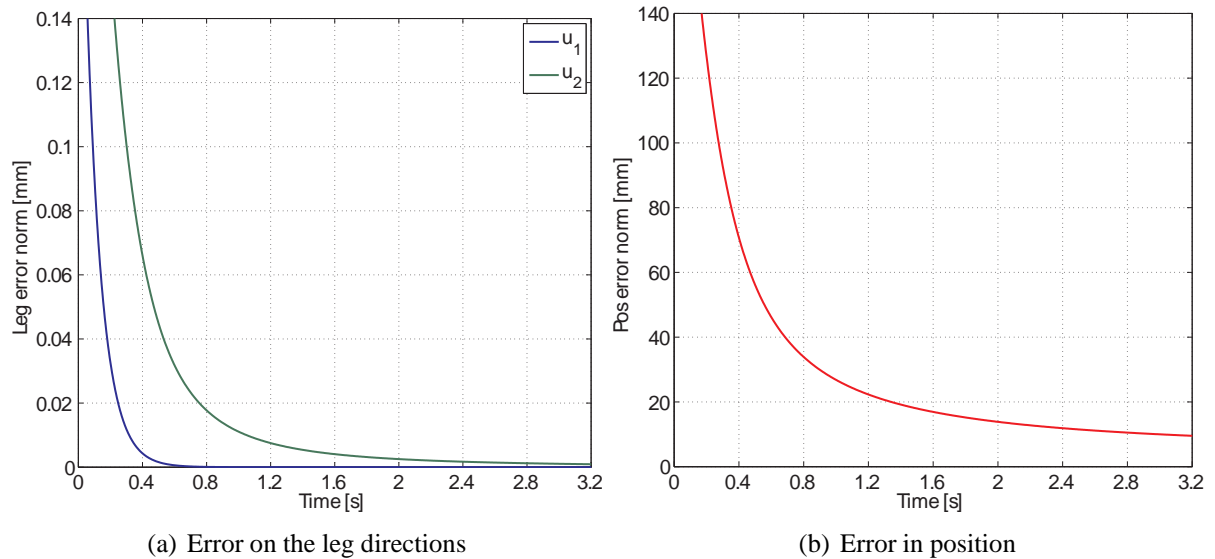


Figure 5.10 – Simulations using the encoder-optimised design.

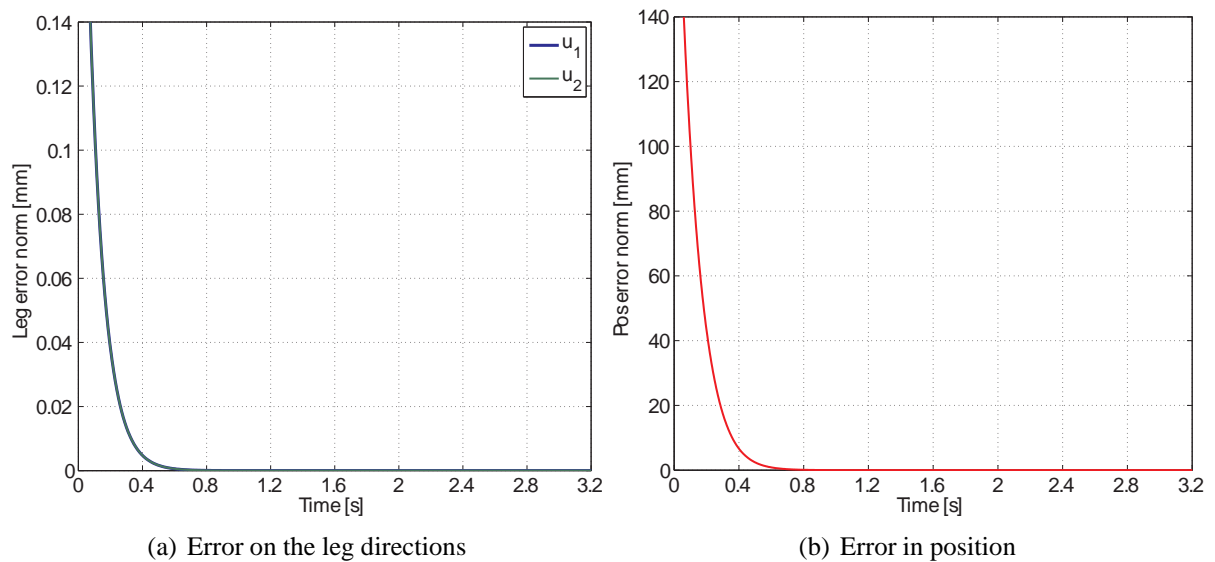


Figure 5.11 – Simulations using the camera-optimised design.

note the norm of the positioning error due to the aforementioned noise in the image, near the desired point  $\{x = -0.15m, y = -0.2m\}$ , which in the case of the encoder-optimised design is near the singularity.

As can be seen in Figure 5.12, the error on the position of the end-effector exceeds the admitted  $0.5mm$ . In the case of the camera-optimised design, this error stay below the threshold (Figure 5.13).

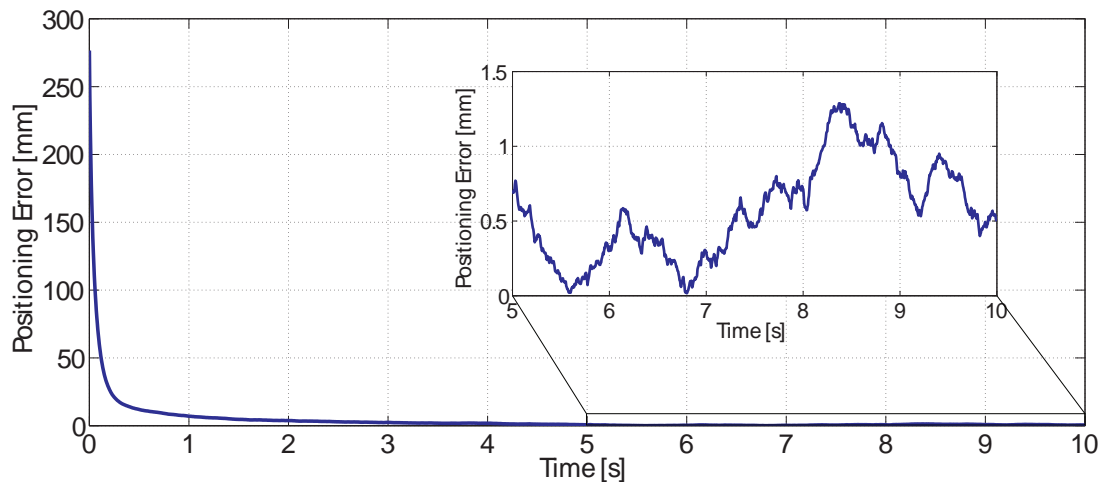


Figure 5.12 – Positioning error at point  $\{x = -0.15m, y = -0.2m\}$  of the encoder-optimised design

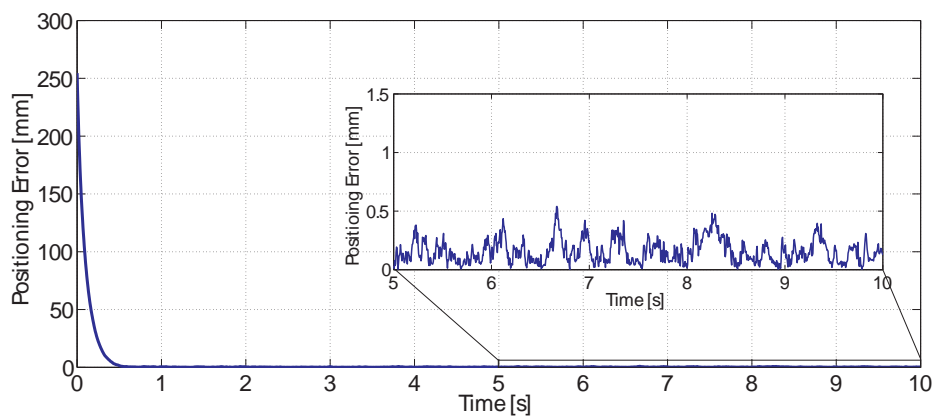


Figure 5.13 – Positioning error at point  $\{x = -0.15m, y = -0.2m\}$  of the camera-optimised design

## 5.6 Conclusions

This chapter presents the utility of the hidden robot concept during the design of robots for tasks when external sensors are used for control.

A five-bar robot architecture is chosen for a pick-and-place task in which the robot should be controlled using observation of its two distal legs. A series of other constraints are presented, in line with the needs of the task. Then, the classical design approach is presented and applied, in order to optimise the geometric parameters of the robot.

The same optimisation process is applied, but the second time, the hidden robot model associated with the given five-bar is also considered within the design process, adding its own constraints. These constraints on the hidden robot actually represent the mapping introduced by the external cameras used in the control, as specified by the task.

When the two final designs are compared, even though the camera-optimised architecture has slightly larger value of the optimisation function to be minimised, it proves to be better suited for the task. The encoder-optimised architecture includes singularities of the hidden robot

within its desired workspace. As a result of this, the encoder-optimised architecture exhibits undesired behaviour during operation (i.e. it does not converge to the desired position).

This proves how important the use of the hidden robot concept is and how it can be used during the early stages of design to more closely relate the concepts of the designer and the needs of the automatician.



## Conclusions

### 6.1 Conclusions

The prevalent use of parallel robots within the industry prompts for ever-increasing performance on the part of robot manufacturers and automaticians who implement them. The classical model-based control schemes are limited due to the complexity of the model, as well as manufacturing and assembly errors that it cannot compensate for appropriately, except through rigorous identification, which can quickly become expensive on a large scale.

The use of external sensors can improve the performance of the robot with respect to the model-based control schemes, through the direct observation/estimation of the end-effector pose, thus bypassing the model entirely. However, the use of external sensors immediately leads to the definition of a virtual robot hidden within the controller, which must be taken into account when designing the real robot for the task, but can also help with the analysis of the controller itself.

The present work presented the hidden robot concept, a tangible visualisation of the mapping between the observation space of the sensors and the Cartesian space of the robot end-effector. Here, we concentrated on the use of external cameras to observe the robot leg edges, from which the leg directions can be extracted, leading finally to an estimation of the robot end-effector. However, it is crucial to note that all the concepts and the entire methodology presented herein are valid for any kind of external sensor observing any kind of robot feature.

The hidden robot model was formalised and generalised through a series of rules which govern the behaviour of the hidden robot, as well as its proper construction, starting from the given real robot. This was first presented on the GS platform, and used to explain certain phenomena which appear when the robot is controlled using leg direction-based vision. After that, it was applied to an Adept Quattro, showing consistent results throughout simulations and experiments.

Afterwards, the hidden robot was used to perform general analyses of different planar and spatial robot families, leading to direct results through the use of tools developed within the mechanical community. Indeed, the hidden robot serves as a powerful tool due to the fact that it permits analysis of accuracy and mapping singularity through simple, easily interpreted, geometric means. In addition, it can be used to perform other tasks relating to the controller, such as providing optimal feature selection. This was demonstrated through different algorithms in simulation on the Adept Quattro, as well as on a special planar mechanism.

Finally, the hidden robot was easily integrated into the geometric optimisation of a five-bar mechanism designed to be used for a visually-controlled pick-and-place task. The hidden robot complimented the real robot during optimisation, by adding its own singularities to the design workspace, which correspond to the singularities of the mapping introduced by the external sensor. The design which did not take into account the hidden robot proved to be less efficient during the performed simulations, albeit scoring better on the optimisation function than the camera-optimised counterpart. The encoder-optimised architecture encountered singularities of the mapping and would not converge to a desired position within the specified workspace. The camera-optimised architecture overcame this, due to the hidden robot eliminating the singularities within the workspace.

Thus, the concept of hidden robot model, associated with mathematical tools developed by the mechanical design community, is a powerful tool able to analyse the intrinsic properties of some controllers developed by the visual servoing community. Due to this overlap of two separate fields which nonetheless need to work together, the hidden robot presents many future research perspectives.

## 6.2 Research Perspectives

At first, experimental validation of the different designs could be envisioned. Though normally it would be difficult to change the design of a robot for comparison, a five-bar mechanism with adjustable link lengths would be fairly easy to construct. Such a mechanism could allow the comparison of many different five-bar designs, using varied constraints, to show that the architectures which take the hidden robot into account are in fact better at performing the given vision-controlled task.

Also, throughout this work, the control law which governed the motion of the robots involved was a simple point-to-point motion. It is conceivable that a complex online trajectory generator could be created using the hidden robot model, such that it would update the trajectory based on algorithms similar to the leg selection algorithm used within this work, in order to achieve the best possible performance in accordance with a given index.

As seen in the experimental data relating to the Adept Quattro, noise can heavily influence the information an external sensor such as a camera receives. A more robust approach therefore would be a hybrid controller, which takes into account both internal and external sensory data. Again, this could be combined in a fashion similar to the leg selection algorithm, to provide



online changing of the weights representing each sensor's contribution to the final control law. This can be implemented not only as a hybrid controller, using internal and external sensors, but also as a controller using several different external sensors. This could even be foreseen during the design step, as was shown within this work, to elaborate different regions in the robot's workspace where the user would clearly know that a particular sensor is dominant. This could easily solve problems such as occlusion, but switching to a different camera angle, or changing to different sensors in extreme light conditions.

The IRSBot-2 is a 2 *dof* planar manipulator with a spatial architecture, which was designed for a high speed pick-and-place task using vision-based control (Germain et al., 2011). For such a high-speed robot, it is important to reflect in detail upon the strategy used for extracting the leg edge data from the image. First of all, a high-speed camera is needed with a high enough frame rate so that the motion of the robot is not extremely high between to adjacent frames. In order to allow faster processing of the data, it is possible to use region of interest extraction from the image, so that only the parts of the image with useful data (i.e. leg edges) are processed. Another approach, depending on the camera placement, is to extract only two full rows of pixels from the image, at the top and bottom, and search for changes in contrast; this would permit finding the points where the leg edges meet the image boundaries and then these could be easily reconstructed with only a fraction of the processing power needed.

Another possible research track is the application of the hidden robot concept to parallel robots whose legs are not cylindrical. It is possible to estimate the pose of an object with an arbitrary shape using classical vision-based techniques. The challenge would be to either assimilate these arbitrary shapes (i.e. the non-cylindrical robot legs) into an equivalent robot with cylindrical legs, and then apply the hidden robot concept as described within this work; or to develop a new strategy for finding the hidden robots associated with these non-cylindrical legs, and see how they behave with respect to the cylindrical-leg hidden robots which we have already studied here.

The hidden robot concept arose from the idea that model-based control methods rely on highly accurate models, without which the controllers do not provide the extreme precision required by certain tasks. As mentioned in Section 2.1.1.2, tree structure robots are becoming more and more widespread. Some of these are quite inaccurate, due to the materials from which they are made, in an attempt to reduce production costs. However, when these robots grasp an object with both arms, they form a closed loop, essentially becoming a parallel robot. It would prove interesting to use the knowledge provided by the hidden robot concept and apply vision-based control to these tree structure robots in the case of multi-arm tasks, in an attempt to increase accuracy when the robot is in 'parallel mode', despite the lower precision in 'tree structure mode'.





## Assembly modes of the studied *ppm* and of their hidden robots

Table A.1: Architectures of the 6 usual *ppm* with 2 *dof*, their assembly modes and their hidden robot models for the visual servoing using leg directions

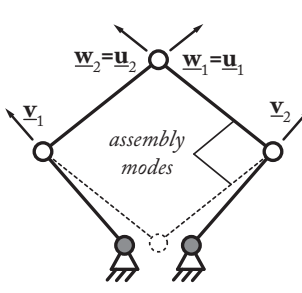
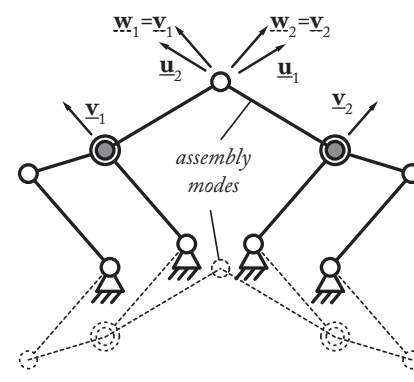
Real robot architecture	Virtual robot architecture
<p><u>RRRRR</u></p> <p>Vertex space of 1 leg: circle</p> <p>Number of solutions to the <i>fkp</i>: 2</p> 	<p><math>\Pi</math><u>RRR</u><math>\Pi</math></p> <p>Vertex space of 1 leg: circle</p> <p>Number of solutions to the <i>fkp</i>: 2</p> 
<p><u>RRRRR</u></p> <p>Vertex space of 1 leg: circle</p> <p>Number of solutions to the <i>fkp</i>: 2</p>	<p><i>idem as for the <u>RRRRR</u> robot</i></p>

Table A.1: Architectures of the 6 usual *ppm* with 2 *dof*, their assembly modes and their hidden robot models for the visual servoing using leg directions

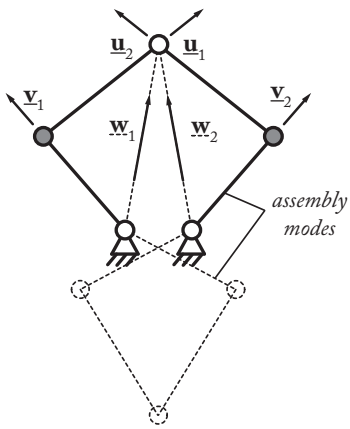
Real robot architecture	Virtual robot architecture
 <p>The diagram illustrates a real robot architecture with two degrees of freedom (DOF). It features a top platform (represented by a circle) and a base (represented by a triangle). The top platform is connected to the base by two legs. The top platform has two legs with joint axes <math>u_1</math> and <math>u_2</math>. The base has two legs with joint axes <math>v_1</math> and <math>v_2</math>. The legs are connected to the base by joints <math>w_1</math> and <math>w_2</math>. The diagram is labeled "assembly modes".</p>	

Table A.1: Architectures of the 6 usual *ppm* with 2 *dof*, their assembly modes and their hidden robot models for the visual servoing using leg directions

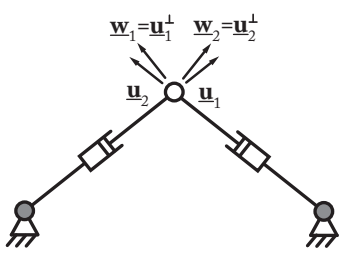
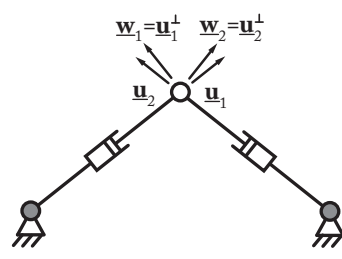
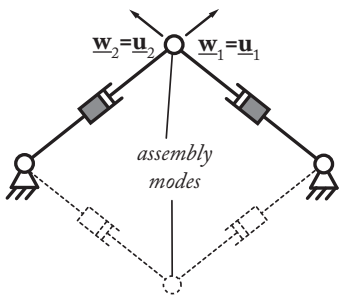
Real robot architecture	Virtual robot architecture
<p><u>RPRPR</u></p> <p>Vertex space of 1 leg: line</p> <p>Number of solutions to the <i>fkp</i>: 1</p> 	<p><u>RPRPR</u></p> <p>Vertex space of 1 leg: line</p> <p>Number of solutions to the <i>fkp</i>: 1</p> 
<p><u>RPRPR</u></p> <p>Vertex space of 1 leg: circle</p> <p>Number of solutions to the <i>fkp</i>: 2</p> 	<p><i>idem as for the <u>RPRPR</u> robot</i></p>

Table A.1: Architectures of the 6 usual *ppm* with 2 *dof*, their assembly modes and their hidden robot models for the visual servoing using leg directions

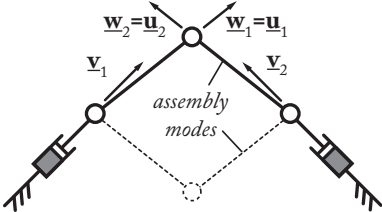
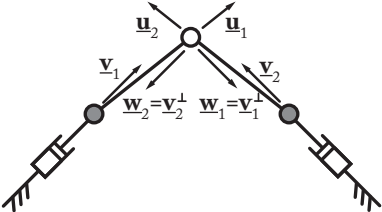
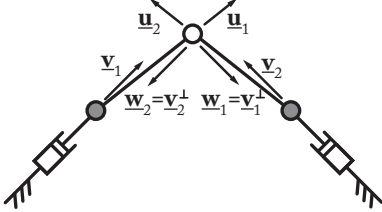
Real robot architecture	Virtual robot architecture
<p><u><i>PRRRP</i></u></p> <p>Vertex space of 1 leg: circle</p> <p>Number of solutions to the <i>fkp</i>: 2</p> 	<p><u><i>PRRRP</i></u></p> <p>Vertex space of 1 leg: line</p> <p>Number of solutions to the <i>fkp</i>: 1</p> 
<p><u><i>PRRRP</i></u></p> <p>Vertex space of 1 leg: line</p> <p>Number of solutions to the <i>fkp</i>: 1</p> 	<p><i>idem as for the <u><i>PRRRP</i></u> robot</i></p>

Table A.2: Architectures of the 6 usual *ppm* with 3 *dof*, their number of assembly modes (outside of singular configurations) and their hidden robot models for the visual servoing using leg directions

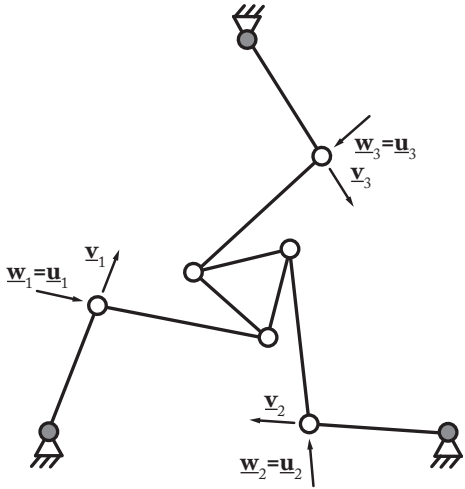
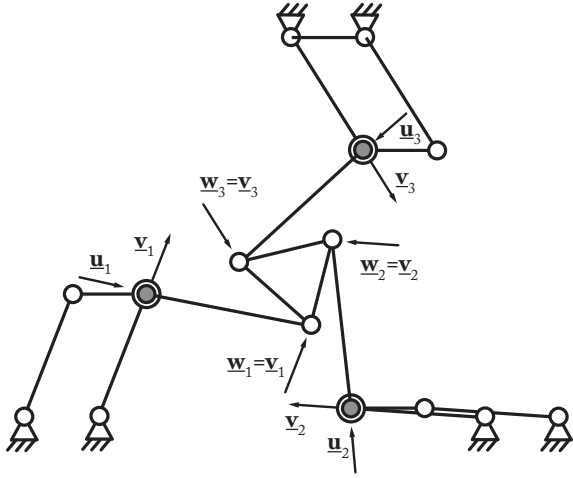
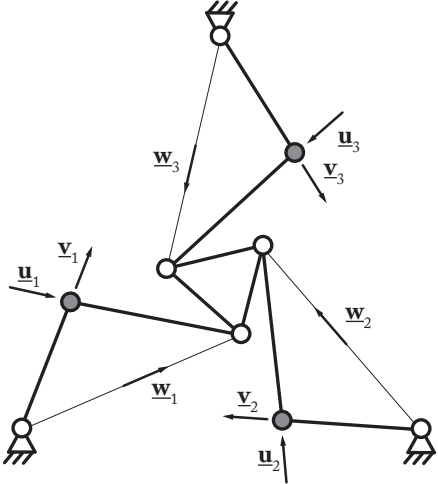
Real robot architecture	Virtual robot architecture
<p>3-<u>RRR</u></p> <p>Vertex space of 1 leg: circle / Coupler curve: sextic</p> <p>Max. number of solutions to the <i>fkp</i>: 6</p> 	<p>3-<u>PIRR</u></p> <p>Vertex space of 1 leg: circle / Coupler curve: sextic</p> <p>Max. number of solutions to the <i>fkp</i>: 6</p> 
<p>3-<u>RRR</u></p> <p>Vertex space of 1 leg: circle / Coupler curve: sextic</p> <p>Max. number of solutions to the <i>fkp</i>: 6</p> 	<p><i>idem as for the 3-<u>RRR</u> robot</i></p>

Table A.2: Architectures of the 6 usual *ppm* with 3 *dof*, their number of assembly modes (outside of singular configurations) and their hidden robot models for the visual servoing using leg directions

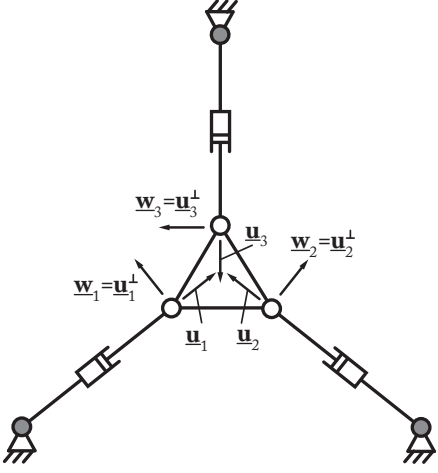
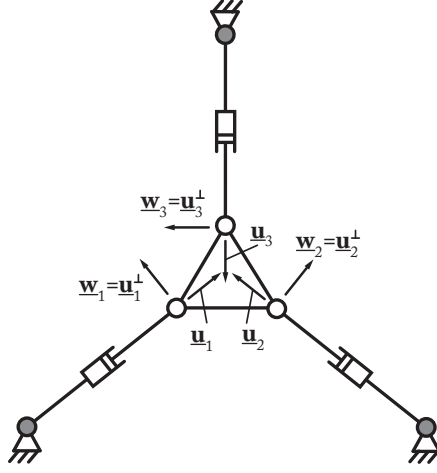
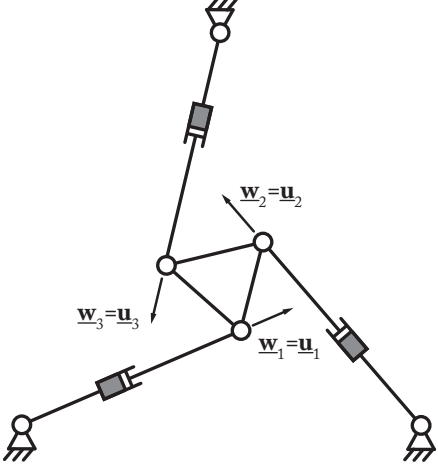
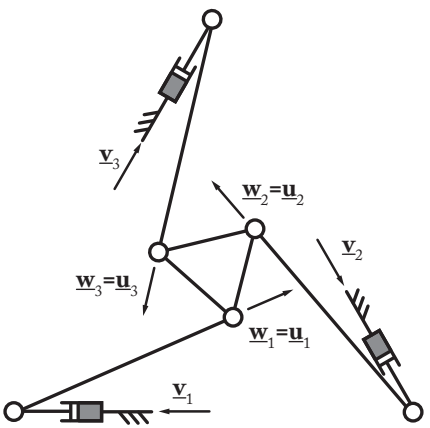
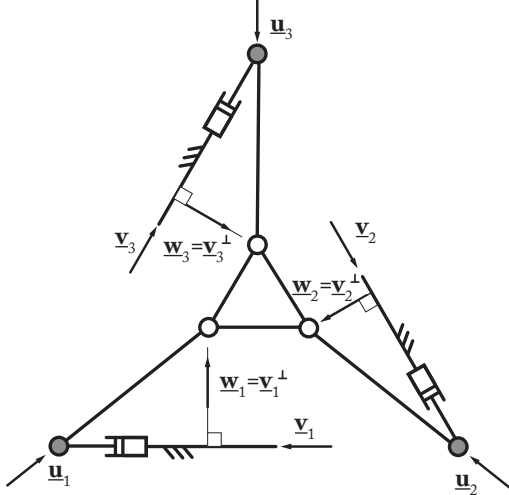
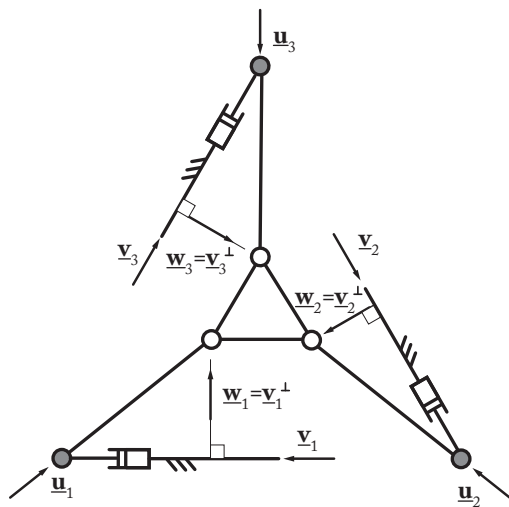
Real robot architecture	Virtual robot architecture
<p>3-<u>RPR</u></p> <p>Vertex space of 1 leg: line / Coupler curve: ellipse</p> <p>Max. number of solutions to the <i>fkp</i>: 2</p> 	<p>3-<u>RPR</u></p> <p>Vertex space of 1 leg: line / Coupler curve: ellipse</p> <p>Max. number of solutions to the <i>fkp</i>: 2</p> 
<p>3-<u>RPR</u></p> <p>Vertex space of 1 leg: circle / Coupler curve: sextic</p> <p>Max. number of solutions to the <i>fkp</i>: 6</p> 	<p><i>idem as for the 3-<u>RPR</u> robot</i></p>



Table A.2: Architectures of the 6 usual *ppm* with 3 *dof*, their number of assembly modes (outside of singular configurations) and their hidden robot models for the visual servoing using leg directions

Real robot architecture	Virtual robot architecture
<p>3-<u>PRR</u></p> <p>Vertex space of 1 leg: circle / Coupler curve: sextic</p> <p>Max. number of solutions to the <i>fkp</i>: 6</p> 	<p>3-<u>PRR</u></p> <p>Vertex space of 1 leg: line / Coupler curve: ellipse</p> <p>Max. number of solutions to the <i>fkp</i>: 2</p> 
<p>3-<u>PRR</u></p> <p>Vertex space of 1 leg: line / Coupler curve: ellipse</p> <p>Max. number of solutions to the <i>fkp</i>: 2</p> 	<p><i>idem as for the 3-<u>PRR</u> robot</i></p>



# Bibliography

- Andreff, N., Dallej, T., and Martinet, P. (2007). Image-based visual servoing of gough-stewart parallel manipulators using legs observation. *International Journal of Robotics Research*, 26(7):677–687. [23](#), [36](#), [37](#), [64](#), [80](#)
- Andreff, N., Espiau, B., and Horaud, R. (2002). Visual servoing from lines. *International Journal of Robotics Research*, 21(8):679–700. [22](#)
- Andreff, N., Marchadier, A., and Martinet, P. (2005). 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. [18](#), [24](#), [25](#), [26](#), [67](#), [80](#)
- Andreff, N. and Martinet, P. (2006). Vision-based kinematic modelling of some parallel manipulators for control purposes. In *Proceedings of EuCoMeS, the First European Conference on Mechanism Science*, Obergurgl, Austria. [72](#)
- Arakelian, V., Briot, S., and Glazunov, V. (2008). Increase of singularity-free zones in the workspace of parallel manipulators using mechanisms of variable structure. *Mechanism and Machine Theory*, 43(9):1129–1140. [32](#)
- Ben-Horin, P. and Shoham, M. (2006). Singularity analysis of a class of parallel robots based on grassmann-cayley algebra. *Mechanism and Machine Theory*, 41(8):958–970. [31](#), [40](#), [64](#)
- Bézout, E. (1764). *Recherches sur le degré des équations résultantes de l'évanouissement des inconnues*. Histoire de l'Académie Royale des Sciences. [44](#)
- Binaud, N., Cardou, P., Caro, S., and Wenger, P. (2010). The kinematic sensitivity of robotic manipulators to joint clearances. In *Proceedings of ASME Design Engineering Technical Conferences*, Montreal, QC, Canada. [32](#)
- Bonev, I. (2002). *Geometric Analysis of Parallel Mechanisms*. PhD thesis, Université Laval, QC, Canada. [72](#), [73](#)
- Bonev, I. (2008). Direct kinematics of zero-torsion parallel mechanisms. In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA 2008)*. [64](#), [72](#)

- Bonev, I., Chablat, D., and Wenger, P. (2006). Working and assembly modes of the Agile Eye. In *Proceedings of the IEEE International Conference On Robotics And Automation (ICRA 1996)*, Orlando, Florida, USA. 68
- Bonev, I., Zlatanov, D., and Gosselin, C. (2003). Singularity analysis of 3-dof planar parallel mechanisms via screw theory. *ASME Journal of Mechanical Design*, 125(3):573–581. 40, 63
- Briot, S. and Arakelian, V. (2008). Optimal force generation of parallel manipulators for passing through the singular positions. *International Journal of Robotics Research*, 27(8):967–983. 67
- Briot, S. and Bonev, I. (2010). Accuracy analysis of 3T1R fully-parallel robots. *Mechanism and Machine Theory*, 45(5):695–706. 54
- Briot, S., Bonev, I., Chablat, D., Wenger, P., and Arakelian, V. (2008). Self motions of general 3-rpr planar parallel robots. *International Journal of Robotics Research*, 27(7):855–866. 62
- Briot, S. and Martinet, P. (2013). 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. 37, 64, 80
- Briot, S., Pashkevich, A., and Chablat, D. (2010). Optimal technology-oriented design of parallel robots for high-speed machining applications. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, Anchorage, Alaska, USA. 84, 87
- Caro, S., Khan, W., Pasini, D., and Angeles, J. (2010a). The rule-based conceptual design of the architecture of serial schonflies-motion generators. *Mechanism and Machine Theory*, 45(2):251–260. 39
- Caro, S., Moroz, G., Gayral, T., Chablat, D., and Chen, C. (2010b). Singularity analysis of a six-dof parallel manipulator using grassmann-cayley algebra and gröbner bases. In *Proceedings of the Symposium on Brain, Body and Machine*, Montreal, QC, Canada. 31, 40, 64, 72, 81
- Carricato, M. and Parenti-Castelli, V. (2002). Singularity-free fully-isotropic translational parallel manipulators. *International Journal of Robotics Research*, 21(2):161–174. 39
- Chablat, D., Moroz, G., and Wenger, P. (2011). Uniqueness domains and non singular assembly mode changing trajectories. In *Proceedings of the 2011 IEEE International Conference on Robotics and Automation (ICRA 2011)*, Shanghai, China. 74
- Chablat, D. and Wenger, P. (2003). Architecture optimization of a 3-dof parallel mechanism for machining applications, the Orthoglide. *IEEE Transactions on Robotics and Automation*, 19(3):403–410. 64, 65, 74

- Chaumette, F. (1998). *The Confluence of Vision and Control*, pages 66–78. Number 237 in LNCIS. Springer-Verlag. [24](#), [27](#)
- Chaumette, F. (2002). *La commande des robots manipulateurs*. Hermès. [18](#)
- Chaumette, F. and Hutchinson, S. (2008). *Handbook of Robotics - Visual Servoing and Visual Tracking*. Springer. [18](#)
- Clavel, R. (1990). Device for the movement and positioning of an element in space. [64](#), [65](#)
- Company, O. and Pierrot, F. (2002). Modelling and preliminary design issues of a 3-axis parallel machine-tool. *Mechanisms and Machine Theory*, 37:1325–1345. [65](#), [72](#)
- Di Gregorio, R. and Parenti-Castelli, V. (1998). A translational 3-dof parallel manipulator. In *Advances in Robot Kinematics: Analysis and Control*, pages 49–58. Springer. [14](#)
- Dombre, E. and Khalil, W. (2010). Modeling, performance analysis and control of robot manipulators. [21](#)
- Espiau, B., Chaumette, F., and Rives, P. (1992). A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3). [17](#)
- French, M. J. and Council, D. (1985). *Conceptual design for engineers*. Springer. [83](#)
- Germain, C., Briot, S., Glazunov, V., Caro, S., and Wenger, P. (2011). Irsbot-2: A novel two-dof parallel robot for high-speed operations. In *Proceedings of the ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Washington DC, USA*. [99](#)
- Germain, C., Caro, S., Briot, S., and Wenger, P. (2013). Optimal design of the irsbot-2 based on an optimized test trajectory. In *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages V06AT07A056–V06AT07A056. American Society of Mechanical Engineers. [87](#)
- Gogu, G. (2004). Structural synthesis of fully-isotropic translational parallel robots via theory of linear transformations. *European Journal of Mechanics. A/Solids*, 23(6):1021–1039. [39](#)
- Gosselin, C. and Angeles, J. (1990). Singularity analysis of closed-loop kinematic chains. *IEEE Transactions on Robotics and Automation*, 6(3):281–290. [12](#)
- Gough, V. and Whitehall, S. (1962). Universal tyre test machine. In *Proceedings of the FISITA 9th International Technical Congress*, pages 117–317. [64](#)
- Guizzo, E. and Ackerman, E. (2012). How rethink robotics built its new baxter robot worker. *IEEE Spectrum*. [7](#)

- Hervé, J. (1992). Group mathematics and parallel link mechanisms. In *Proceedings of the IMACS/SICE International Symposium. on Robotics, Mechatronics, and Manufacturing Systems*, pages 459–464, Kobe, Japan. [75](#)
- Honegger, M., Brega, R., and Schweitzer, G. (2000). Application of a nonlinear adaptive controller to a 6 dof parallel manipulator. In *Proceedings of the IEEE ICRA*, pages 1930–1935, San Francisco, CA, USA. [16](#)
- Honegger, M., Codourey, A., and Burdet, E. (1997). Adaptive control of the Hexaglide, a 6 dof parallel manipulator. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 1997)*. [65](#)
- Horaud, R. and Dornaika, F. (1995). Hand-eye calibration. *The international journal of robotics research*, 14(3):195–210. [22](#)
- Horaud, R., Dornaika, F., and Espiau, B. (1998). Visually guided object grasping. *IEEE Transactions on Robotics and Automation*, 14(4):525–532. [17](#)
- Hunt, K. H. (1978). *Kinematic geometry of mechanisms*. Clarendon Press Oxford. [14](#)
- Khalil, W. (2012). Advanced modelling of robots. Lecture. École Centrale de Nantes. [10](#)
- Khalil, W. and Dombre, E. (2002). *Modeling, Identification and Control of Robots*. Hermes Penton London. [33](#)
- Kock, S. and Schumacher, W. (2000). A mixed elastic and rigid-body dynamic model of an actuation redundant parallel robot with high-reduction gears. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 2, pages 1918–1923. IEEE. [16](#)
- Kong, X. and Gosselin, C. (2002). A class of 3-dof translational parallel manipulators with linear input-output equations. In *Proceedings of the Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 3–4, Québec City, QC, Canada. [39](#)
- Krut, S., Company, O., Benoit, M., Ota, H., and Pierrot, F. (2003). I4: A new parallel mechanism for Scara motions. In *Proceedings of the 2003 International Conference on Robotics and Automation (ICRA 2013)*. [65](#), [72](#)
- Leinonen, T. (1991). Terminology for the theory of machines and mechanisms. *Mechanism and Machine Theory*, 26. [1](#)
- Ma, O. and Angeles, J. (1992). Architecture singularities of parallel manipulators. *The International Journal of Robotics and Automation*, 7(1):23–29. [32](#)
- Malis, E., Chaumette, F., and Boudet, S. (1999). 21/2d visual servoing. *Robotics and Automation, IEEE Transactions on*, 15(2):238–250. [20](#), [21](#)

- Martinet, P., Gallice, J., and Khadraoui, D. (1996). 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. [17](#), [29](#)
- Merlet, J. (2006a). Jacobian, manipulability, condition number, and accuracy of parallel robots. *ASME Transactions Journal of Mechanical Design*, 128(1):199–206. [10](#), [32](#)
- Merlet, J. (2006b). *Parallel Robots*. Springer, 2nd edition. [1](#), [22](#), [23](#), [25](#), [31](#), [39](#), [40](#), [59](#), [61](#), [62](#), [64](#), [66](#), [72](#), [73](#), [74](#)
- Merlet, J.-P. (2004). Solving the forward kinematics of a gough-type parallel manipulator with interval analysis. *The International Journal of robotics research*, 23(3):221–235. [10](#)
- Michel, H. and Rives, P. (1993). Singularities in the determination of the situation of a robot effector from the perspective view of 3 points. Technical report, INRIA. [67](#), [80](#)
- Nabat, V., de la O Rodriguez, M., Company, O., Krut, S., and Pierrot, F. (2005). Par4: very high speed parallel robot for pick-and-place. In *Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*. [8](#), [64](#), [65](#), [74](#)
- Oen, K.-T. and Wang, L.-C. T. (2007). Optimal dynamic trajectory planning for linearly actuated platform type parallel manipulators having task space redundant degree of freedom. *Mechanism and machine theory*, 42(6):727–750. [16](#)
- Paccot, F., Andreff, N., and Martinet, P. (2009). A review on the dynamic control of parallel kinematic machines: Theory and experiments. *The International Journal of Robotics Research*, 28(3):395–416. [16](#)
- Pashkevich, A., Chablat, D., and Wenger, P. (2009). Stiffness analysis of overconstrained parallel manipulators. *Mechanism and Machine Theory*, 44(5):966–982. [32](#)
- Pierrot, F., Uchiyama, M., Dauchez, P., and Fournier, A. (1990). A new design of a 6-dof parallel robot. *Proceedings of the 23rd International Symposium on Industrial Robots. Journal of Robotics and Mechatronics*, pages 308–315. [65](#)
- Plücker, J. (1865). On a new geometry of space. *Philosophical Transactions of the Royal Society of London*, 155:725–791. [22](#)
- Rosenzweig, V., Briot, S., Martinet, P., Ozgur, E., and Bouton, N. (2014). 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. [80](#)
- SiRoPa Toolbox (2015). [74](#)
- Swevers, J., Ganseman, C., Tukel, D., DeSchutter, J., and VanBrussel, H. (1997). Optimal robot excitation and identification. *IEEE Transactions on Robotics and Automation*, 13:730–740. [16](#)

- Tale Masouleh, M., Gosselin, C., Husty, M., and Walter, D. (2011). Forward kinematic problem of 5-RPUR parallel mechanisms (3T2R) with identical limb structures. *Mechanism and Machine Theory*, 46:945–959. [44](#)
- Thuilot, B., Martinet, P., Cordesses, L., and Gallice, J. (2002). Position based visual servoing: keeping the object in the field of vision. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1624–1629. IEEE. [20](#)
- Tischler, C., Hunt, K., and Samuel, A. (1998). A spatial extension of cardanic movement: its geometry and some derived mechanisms. *Mechanism and Machine Theory*, 33:1249–1276. [30](#), [31](#)
- Tsai, L. (1999). *Robot Analysis: The Mechanics of Serial and Parallel Manipulators*. Wiley-Interscience publication. John Wiley & Sons. [8](#)
- Tsai, L. (2000). Kinematics and optimization of a spatial 3-upu parallel manipulator. *ASME Journal of Mechanical Design*, 122:439–446. [64](#)
- Tsai, L. and Joshi, S. (2001). Comparison study of architectures of four 3 degree-of-freedom translational parallel manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2001)*. [65](#)
- Tsai, R. Y. and Lenz, R. K. (1989). A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *Robotics and Automation, IEEE Transactions on*, 5(3):345–358. [22](#)
- Vignolo, A. (2014). Visual servoing of the Monash epicyclic-parallel manipulator. Master's thesis, École Centrale de Nantes. [81](#)
- Vignolo, A., Briot, S., Martinet, P., and Chen, C. (2014). Comparative analysis of two types of leg-observation-based visual servoing approaches for the control of the five-bar mechanism. In *Australasian Conference on Robotics and Automation (Denny Oetomo 2 December 2014 to 4 December 2014)*, pages 1–10. Australian Robotics and Automation Association. [80](#)
- Vivas, A., Poignet, P., Marquet, F., Pierrot, F., and Gautier, M. (2003). Experimental dynamic identification of a fully parallel robot. In *Proceedings of the 2003 IEEE International Conference on Robotics & Automation (ICRA 2003)*, Taipei, Taiwan. [16](#)
- Wolf, A., Shoham, M., and Park, F. (2002). Investigation of singularities and self-motions of the 3-UPU robot. In *Advances in Robot Kinematics*, Dordrecht, Germany. [64](#)
- Zein, M., Wenger, P., and Chablat, D. (2008). Non-singular assembly-mode changing motions for 3-RPR parallel manipulators. *Mechanism and Machine Theory*, 43(4):480–490. [67](#)
- Zlatanov, D., Bonev, I., and Gosselin, C. (2002). Constraint singularities of parallel mechanisms. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2002)*. [14](#)



---

Zlatanov, D., Fenton, R. G., and Benhabib, B. (1994). Singularity analysis of mechanisms and robots via a motion-space model of the instantaneous kinematics. In *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 980–985. IEEE.





# Thèse de Doctorat

**Victor ROSENZVEIG**

**Conception et commande référencées capteurs de robot rapides**

**Sensor-Based Design and Control of High-Speed Manipulators**

## Résumé

Les manipulateurs parallèles ont des modèles très complexes et les contrôleurs classiques utilisés sont sensibles aux erreurs de modélisation. Afin de supprimer ces erreurs, il est possible d'utiliser des capteurs extéroceptifs pour mesurer la pose de l'effecteur. Les caméras peuvent être utilisées, offrant une précision plus élevée que dans le cas de contrôleurs à base de modèles. Dans certains cas, il est impossible d'observer directement l'effecteur, et les directions des jambes du robot peuvent être observées à la place. Toutefois, cela conduit à des problèmes inhabituels de non-convergence. Ces résultats peuvent être expliqués par le concept de robot caché, qui est une visualisation concrète de la cartographie entre l'espace des directions des jambes observées et l'espace cartésien. Dans ce manuscrit on présente la formalisation et la généralisation du concept de robot caché. Ce concept est ensuite validé sur un Adept Quattro, par des simulations et des expériences, ce qui prouve son utilité en termes d'analyse de la convergence et de la précision, ainsi que pour trouver les singularités et les minima locaux dans la cartographie. Ensuite, de nouvelles applications basées sur le robot caché sont développées. Cela permet une analyse de contrôlabilité pour différents familles de robots plans et spatiaux en utilisant des méthodes géométriques simples. Ensuite, on fournit des algorithmes de sélection de l'ensemble optimal de jambes à observer. Enfin, le robot caché est utilisé pour la conception basée capteurs, en intégrant les besoins du système de contrôle basé capteurs dans le processus de conception, en particulier pour l'optimisation des paramètres géométriques d'un mécanisme à cinq barres.

## Mots clés

Robots parallèles, Commande par vision, Conception basée capteur, Robot caché.

## Abstract

Parallel manipulators have very complex models and the classical control schemes used are susceptible to modelling errors. In order to suppress these errors, it is possible to use exteroceptive sensors to measure the end-effector pose directly. Cameras can offer this opportunity, providing higher accuracy than in the case of model-based control schemes. In some cases, it is impossible to directly observe the end-effector, and the robot leg directions can be observed instead. However, this leads to unusual problems of non-convergence. These results can be explained through the use of the hidden robot concept, which is a tangible visualisation of the mapping between the observed leg direction space and the Cartesian space. This manuscript offers a formalisation and generalisation of the hidden robot concept. It is then validated on an Adept Quattro, through simulations and experiments, proving its usefulness in terms of convergence and accuracy analysis, as well as finding singularities and local minima within the mapping. Then, further applications based on the hidden robot concept are developed. This allows for a controllability analysis of different planar and spatial robot families using simple geometrical methods. Then, we provide algorithms for selecting the optimal set of legs to be observed, as well as providing means for optimal feature selection. Finally, the hidden robot concept is used as a tool for sensor-based design, by integrating the needs of the sensor-based control scheme into the design process, specifically for optimisation of the geometric parameters of a Five-Bar mechanism.

## Key Words

Parallel robots, Vision-based control, Sensor-based design, Hidden robot.