Contributions to Visual Servoing for legged and linked multicomponent robots

By

Zelmar Echegoyen Ferreira

http://www.ehu.es/ccwintco

Dissertation presented to the Department of Computer Science and Artificial Intelligence in partial fulfillment of the requiremets for the degree of
Doctor of Philosophy

PhD Advisor:

Prof. Alicia d'Anjou d'Anjou

Prof. Manuel Graña Romay

At
The University of the Basque Country
Donostia - San Sebastian
2009

Universidad del País Vasco Euskal Herriko Unibertsitatea

AUTORIZACION DEL/LA DIRECTOR/A DE TESIS

PARA SU PRESENTACION

Dr/a. _____con N.I.F._____

como Director/a de la Tesis Doctoral: _____

_____

_____

realizada en el Departamento _____

_____

por el Doctorando Don/ña. _____ ,

autorizo la presentación de la citada Tesis Doctoral, dado que reúne las condiciones necesarias para su defensa.

En _____a _____de _____de _____

EL/LA DIRECTOR/A DE LA TESIS

Fdo.: _____

Universidad del País Vasco Euskal Herriko Unibertsitatea

CONFORMIDAD DEL DEPARTAMENTO

El Consejo del Departamento de _____

_____

en reunión celebrada el día ____ de _____ de _____ ha acordado dar la conformidad a la admisión a trámite de presentación de la Tesis Doctoral titulada: _____

_____

_____

dirigida por el/la Dr/a. _____

_____

y presentada por Don/ña. _____

ante este Departamento.

En _____a _____de _____de _____

Vº Bº DIRECTOR/A DEL DEPARTAMENTO SECRETARIO/A DEL DEPARTAMENTO

Fdo.: _____ Fdo.: _____

Universidad del País Vasco Euskal Herriko Unibertsitatea

ACTA DE GRADO DE DOCTOR

ACTA DE DEFENSA DE TESIS DOCTORAL

DOCTORANDO DON/ÑA. _____

TITULO DE LA TESIS: _____

_____

_____

El Tribunal designado por la Subcomisión de Doctorado de la UPV/EHU para calificar la Tesis Doctoral arriba indicada y reunido en el día de la fecha, una vez efectuada la defensa por el doctorando y contestadas las objeciones y/o sugerencias que se le han formulado, ha otorgado por_____la calificación de:

unanimidad ó mayoría

En _____a _____de _____de _____

EL/LA PRESIDENTE/A, EL/LA SECRETARIO/A,

Fdo.: Fdo.:

Dr/a: _____ Dr/a: _____

VOCAL 1º, VOCAL 2º, VOCAL 3º,

Fdo.: Fdo.: Fdo.:

Dr/a: _____Dr/a:_____Dr/a:_____

EL/LA DOCTORANDO/A,

Fdo.: __

Agradecimientos

En primer lugar me gustaría agradecer a mis directores de tesis, Prof. Don Manuel Graña Romay y Prof. Doña Alicia d'Anjou d'Anjou, por la motivación y guía que han realizado de mi trabajo.

También quiero agradecer a mis compañeros de laboratorio por la ayuda y el apoyo que han significado en momentos difíciles.

A Deysi y Tximi por la paciencia y comprensión que han tenido durante este último tiempo.

Muchas gracias,

Zelmar

Contributions to Visual Servoing for legged and linked multi-component robots by

Zelmar Echegoyen Ferreira

Submitted to the Department of Computer Science and Artificial Intelligence on October 21, 2009,
in partial fulfillment of the requirements for the degree of Doctor of Philosophy

Abstract

This PhD Thesis contains two main contributions to the field of robotics and Visual Servoing: a principled approach to the Visual Servoing of legged robots, and contributions to the modeling, control and Visual Servoing of Linked Multi-Component Robotic Systems (MCRS). We have also performed a comprehensive review on Visual Servoing. For legged robots we have developed a formal and rigorous construction of the image Jacobian of a generic legged robot, based on the minimization of the visual error and taking into account all the degrees of freedom of the robot. We have specialized it to the the Sony's Aibo ERS-7 robot, building the implementation of the control on the robot. We have done a systematic empirical experiment to asses the model application range and its sensitivity. The Linked MCRS consists on a group of robots carrying a passive uni-dimensional object (hose or wire). To our knowledge this is the first formal study of such a system. We have built a model of the system dynamics based on dynamic splines that allows the simulation of the system, including heuristic control algorithms for the robots. This model allows the study of the effect of several hose parameters, such as its weight and rigidity, and the robots positions. We have also derived analytically from this model the inverse kinematics for the motion of the hose from an initial to a desired configuration. Finally, we have done the physical realization of centralized visual control experiments of a Linked MCRS, with a group of SR1 robots carrying a relatively rigid electric wire, which is also the first reported attempt to realize such kind of systems.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter gives a fast overview of the PhD Dissertation Report. In section 1.1 we give some motivation of our work. In section 1.2 we state the pursued objectives while in section 1.3 we highlight the contributions of our work to the current state of the art. Finally, section 1.4 gives a guide of the elements of this document.

1.1 Motivation

The introduction of robotic systems in all aspects of the industry has been growing steadily for the last 30 years, and the demands of the industry have open new research areas in robotics. Nowadays the frontiers for the industrial robotic systems are pushing towards highly unstructured environments and, therefore, innovative approaches must be developed in order to face these new challenges.

This PhD work has two main branches. The first one concerns Visual Servoing and the study of a visual control strategy for the tracking of objects by legged robots, comprising a review of Visual Servoing done in chapter 2 to the construction of the image Jacobian matrix for the Sony's Aibo robot done in chapter 3. The second branch is the application of multi-robots heuristics for the transport of a uni-dimensional object, as a hose o wire, by a group of cooperative robots in chapter 4.

1.1.1 Visual Servoing on the Aibo

In the industrial environments the sensorization has been done traditionaly by a fixed parametrization of the sensors and the control process, performing manually the system fine tuning and even adapting the environment to fit the requirements of predetermined tasks. In order to use robots outside of a controlled environment, more sofisticated sensor devices and control strategies must be employed, hence the study of the vision systems as the sensing subsystem for robot systems has acquired great importance as a research field, since vision systems allow to get a comprehensive description of the environment, offering huge amounts of information and it is a remote and non intrusive sensor. Vision systems promise to be specially useful in low structured environments where the environment features are constantly varying. The systems that use a visual control in a closed loop do not need to know beforehand the exact structure of the environment and the position of the robot articulations, because they can compensate the deviations through the visual feedback. However, the visual feedback needs a high communication bandwidth and computing power for high frequency image processing. Among these approaches, the collection of techniques and approaches known as Visual Servoing has had a significant increase in developments and application in last years.

Visual Servoing can be broadly defined as the task of positioning one or more robots in order to get the desired poses of their final effectors, using as feedback input to the control loop the estimated positioning error computed from the visual information extracted of the environment by one or more video-cameras. The pose of the final effector is defined as the position and orientation of the last element of its chain of articulations. For mobile robots Visual Servoing refers to the robot pose relative to some landmarks detected in the environment.

Visual Servoing has grown as a discipline having a strong fundamental formalization. The formal approach aims to the analytical derivation of the inverse kinematics of the robot from the quantitative visual error. To this end, it is precise to be able to formulate the direct kinematics that relate the camera, and consequently the image viewed, to the robot's degrees of freedom. Because of the great complexity of the systems and the existence of some (implicit) non-linearities, the approaches found in the literature always resort to some kind of linearization or reduced linear model, that can be inverted analytically. By this same reason, it is necessary to asses the

extent of the robot's behavior space that is well approached by the proposed kinematic model and its inversion. This assessment must be performed on real implementations on real robots performing the task physically.

Traditionally, mobile robots make use of wheels to move around, which is obviously limited to planar surfaces. Due to this limitation new kinds of robot locomotion have been proposed, among them legged robots have been proposed in several architectures. Vision based control approaches and solutions for this kind of robots following the Visual Servoing approach have been scarce in the literature, with most of the approach only moving the effectors linked directly to the camera or using a high level control commands without a direct link between the visual error and the basic robot kinematics. In this work we aim to follow a principled approach building a detailed model of the image Jacobian matrix which formalizes a linear approach to the robot's kinematics, taking into account all the effectors that can affect the image captured by the robot's camera.

Amongst the legged robots the Sony's Aibo robot has acquired great importance, mainly by its commercial success, although its production has been discontinued by Sony for its own strategic reasons. It reached such a high level of acceptance in the robotics community that there was a specific league in the RoboCup robot soccer championship. Many teams participating in this competition have implemented some Visual Servoing approaches [59, 61] in the Aibo robot to track the ball. However, these approaches are usually limited to the movement of the head effectors in order to keep the ball inside the video image. Our aim in this PhD work is the development of a more inclusive visual control strategy for the tracking of the ball involving all the degrees of freedom of the robot. We will also want to explore the robustness and range of validity of such a principled approach in real life realizations. Because random experiments give little information in this regard, we have tried to perform a systematic experiment to explore this issue.

1.1.2 Towards Visual Servoing on Linked Multi-Component Robotic Systems

On the other hand, when the robotic tasks are non repetitive, placed in a non structured and highly dynamic environment, the use of teams of robots may be required, each of the members of the team can be specialized in a specific task, maybe controlled using traditional or innovative robotics approaches.

The versatility of robots implies a greater ability in them to cooperate with each other in order to execute a task. It is in this context where multi-robots systems are getting more attention and constituting a new research area. In a recent review [15] some kinds of multi-component robotic systems (MCRS) have been categorized and analyzed from several points of view. There, a distinction among Modular, Linked and Distributed MCRS was made. This distinction is based on morphological features, on the way that the component robots are interrelated physically. Morphology has also strong influence on the system's functionality. Therefore, these different kinds of systems are better suited for different kinds of tasks. Distributed MCRS are better suited for exploration and distributed sensing, Modular MCRS are better suited for tasks that require morphological adaptations, such as changes to adapt to strong variations of terrain, like the aggregations performed by the swarm-bots to pass over trenches. Linked MCRS are the natural way to perform the transportation of uni-dimensional objects like hoses or wires.

In some highly unstructured working environments, like shipyards or construction sites, one of the most frequently required operations is the deployment and manipulation of hoses, power-lines, and the like, that is, unidimensional objects that serve for the transportation of fluids or power. The automatic deployment, manipulation, transportation and collection of such items poses a broad avenue for research. Here, in this PhD, we are only scratching the surface of the problem. Among the issues that can be identified are:

- Self-sensing: the ability of the system to perceive its own status, where the robots are placed or how the hose is deployed.
- Adaptive control: the ability to change the programmed behavior in order to adapt to changes in the environment or the system's state, including failure recovery.
- Distributed sensing: because the hose can be traversing separate regions of the environment, with different properties, the problem of fusing all this information on a manageable representation of the world taking into account also the interactions induced by the hose itself, becomes an interesting issue.

In this PhD, we have to be modest and limit ourselves to some realistically addressable issues. We have been interested in the formal modeling of the

MCRS-hose system from a geometrical and dynamical point of view. This kind of models are useful for the realistic simulation of the system that may allow to predict or analyze its behavior under different control strategies, which can be heuristic or formally derived. Besides, the formal model allows also the formal derivation of control strategies by the inversion of the forward kinematic models. We have devoted some efforts in this respect, because we think that this kind of control strategies could be applied once several identification problems have been solved.

The formal and simulated works have little meaning unless we can test physically (1) that the problem really exists and the approach itself has some merit, (2) that a physical demonstration exists. Therefore, some efforts have been devoted to build and test a physical prototype. Testing ideas at the physical level has a big disadvantage from the academic point of view: most of the work has little academic innovation. Building the robots, tuning the communication links or the image processing algorithm, ensuring that the experiment runs from beginning to end without interferences... All this work will remain outside of this PhD dissertation, haunting the reader with its untold burden.

1.2 Objectives

In this section we will enumerate the objectives set for the PhD work. We will distinguish between operative and scientific objectives. The former understood as preliminar or necessary work for the later. These objectives have been reached to some extent, although in some respects much remains to be done.

The scientific objectives of this work are the following ones:

- To extend the application of Visual Servoing to two new instances
  - The full control of all degrees of freedom of a legged robot from visual feedback, including legs' degrees of freedom.
  - The control of Linked MCRS. For such kind of systems, the long term objective is the proposition of distributed control processes. A more close to the ground objective is the realization of a centralized control, where perception and control lie in a central processing unit.

- To perform an empirical assessment of the validity of the Visual Servoing approaches testing its limitations and behavior in real life realizations and under realistic conditions.
- To provide a model of Linked MCRS that could be easily adapted to new instances of the system, varying the parameters of the linking element and/or the individual robots. This model would be used to:
  - Simulate the system: system simulation may allow to understand its dynamics to explain some phenomena found in the real life experimentation, or to predict interesting/undesired behaviors of the whole system.
  - Test control strategies: whatever the source of the strategy definition, and specially for heuristic algorithms, it is desirable to be able to determine its effects in simulation environments before going into its physical realization.
- To derive control strategies from the analytical model of the Linked MCRS.
- To demonstrate empirically the fundamental differences between Distributed and Linked MCRS. The interaction with the passive linking element (the hose) may introduce some effects that can not possibly happen on Distributed system. Therefore, strategies such as follow-the-leader have a quite different realization from one kind of system to the other.

The operational objectives that we had to pursuit to achieve or, simply, attack the scientific objectives are the following:

- Build a detailed kinematic model of the Aibo from the available descriptions.
- Build a software development environment for the Aibo using available SDK and computing equipment.
- Design a systematic Visual Servoing assessment experiment on the Aibo, including the design of the environment, the software for the measurement and analysis of the results and their management.

- Realization of the Visual Servoing assessment experiment with the Aibo, with took many hours and several replications due to miscellaneous errors and problems.
- Adaptation of the Geometrically Exact Dynamic Splines model [76] to the MCRS problem, including bits of dynamical physical modeling.
- Programming in Matlab the realization of the Linked MCRS simulation based on spline models.
- Realization of simulation experiments to reproduce an interesting phenomenon (the loop in the hose), to seek strategies/conditions to minimize it, and to test actual heuristic control strategies.
- Video recording, composition, editing and watermarking for its publication in the group's wiki page of the various results obtained in the thesis.
- For the construction of the real life Linke MCRS experiment we had to[1]
  - Build the robots, that came as a didactic toolkit.
  - Test the communications, evaluate them and redesign the communication protocol for robust communication between the central processor and the robots.
  - Programming and Testing the image processing algorithm until reaching good parameter tuning, including the task of painting the robots with a better suited color (both floor and robots were yellow).
  - Design the heuristic control algorithm, implement it in the central processing and test it in the actual system.

1.3 Contributions of the PhD Thesis

We have performed a review of the state of the art of Visual Servoing that sets the stage for much of the work described in this PhD dissertation. After that

[1]This part of the thesis work has been done in close collaboration with Ivan Villaverde. Ramon Moreno worked on the image processing algorithm.

we have to state separately the contributions corresponding to the two main research lines explored in this Thesis: Visual Servoing of a Legged Robot described in chapter 3, and the works on Linked MCRS described in chapter 4. Later we enumerate the publications related to the thesis work and the research projects at which the PhD candidate has somewhat contributed.

1.3.1 Contributions to Visual Servoing of Legged Robots

Our general approach to the Visual Servoing of legged robots follows the formal lines of the Image Based Visual Servoing (IBVS) systems described in chapter 2.

- We build a locally linear kinematic model of the robot by composing the diverse Jacobian matrices that embody the dependencies among observation and control parameters. Then we propose a simple inversion of the model to obtain the desired control commands that will accomplish the minimization of the perceptual error detected in the vision system. The inversion performed is robust against singularities due to under/over constrained systems.

- One of the key problems in the development of this approach is the definition of a ground reference system needed to relate the effect of the articulations on the perception of the objects in the world. This ground reference system is trivial for static manipulator robots, however it can be arbitrary and changing for legged and mobile robots. We have solved the problem by using the tips of the legs that are the ground contact points to define this ground reference system. Another basic problem was the determination of the ground plane upon which the robot is resting. We have solved it by using the joint state information provided by the robot basic control systems.

- We have specialized the general approach to the Aibo robot, performing the actual implementation of entire approach on the on-board robot computer, where we obtained real time processing from the Aibo's camera. In other words, we have developed a real time demonstrator of the approach.

- We have performed a systematic empirical assessment of the Visual Servoing performance on the Aibo. Video samples show the real time

performance of the system. Plots of the visual error show that the system reach low errors whose distribution is invariant to the distance of the tracked object (the ball) to the camera. The realization of a sequence of Visual Servoing processes changing the ball position shows further robustness as the final error remains similar at each step of Visual Servoing.

1.3.2 Contributions on Linked MCRS development and Visual Servoing

The following are the contributions of this thesis to the state of the art in Linked MCRS:

- We have built a geometrical and dynamical model of the Linked MCRS based on the formalism of Geometrically Exact Dynamic Splines. This model is parametrized by the physical properties of the passive linking element (the hose).
- We have built an test a simulation environment for this kind of models. We reproduced an observed phenomenon using this simulation system.
- We have derived an inverse kinematics/dynamics approach to the control of the robots in order for the whole system to achieve a desired configuration. We have tested it simulation studies.
- We have tested several approaches for the transportation of the hose in simulation studies.
- We have actually realized a linked system showing behaviors that clearly separate Linked MCRS from other kinds of MCRS, thus, opening a whole area of research. Some interesting features of this physical realization:
  - It implements a centralized Visual Servoing akin to a Position Based Visual Servoing (PBVS) using a robust image segmentation algorithm.
  - The individual robots are programmed as independent agents, though computing their behavior lies in the central computer, knowing only the hose segment ahead of them,

1.3.3 Publications

- Z. Echegoyen, A. d'Anjou and M. Graña. Modeling a legged robot for visual servoing. In Computational Science and its Applications - ICCSA 2007, volume 4707/2007, pages 798-810, 2007.

- I. Villaverde, Z. Echegoyen and M. Graña. Neuro-evolutive system for egomotion estimation with a 3d camera. Australian Journal of Intelligent Information Processing Systems, 10(1):59-70, 2008.

- Z. Echegoyen, A. d'Anjou and M. Graña. Contribution to legged robot visual servoing. In Knowledge-Based Intelligent Information and Engineering Systems, pages 1179-1186, 2009.

- Z. Echegoyen, A. d'Anjou and M. Graña. On the control of a multirobot system for the manipulation of an elastic hose. In Bioinspired Applications in Articial and Natural Computation, pages 382-389, 2009.

- Z. Echegoyen, A. d'Anjou, I. Villaverde and M. Graña. Towards the adaptive control of a multirobot system for an elastic hose. In Advances in Neuro-Information Processing, pages 1045-1052, 2009.

- I. Villaverde, Z. Echegoyen and M. Graña. Neuro-Evolutive system for Ego-motion estimation with a 3D camera. In Advances in Neuro-Information Processing, pages 1021-1028, 2009.

- Z. Echegoyen, I. Villaverde, R. Moreno, M. Graña and Alicia d'Anjou. Linked mobile robot control: the hose manipulation problem. Robotics and Autonomous Systems (In preparation, estimated submission November 2009).

- Z. Echegoyen, A. d'Anjou and M. Graña. Visual servoing on a legged robot: formal modelling and empirical validation. IEEE Transactions on Robotics (In preparation, estimated submission December 2009).

1.3.4 Research projects

- Percepción artificial y control de caos para robótica modular en entornos dinámicos y no estructurados (McRobs), Ministerio de Educación y Ciencia, 2006. Investigador Principal: Manuel Graña Romay.

Entidades colaboradoras: Ciencias de la Computación e Inteligencia Artificial (UPV/EHU), Universidad de A Coruña, Inteligencia Artificial (UPM). Duración: 36 meses. Referencia: DPI2006-15346-C03-03.

1.4 Structure of the PhD Dissertation report

The PhD report has the following structure:

1. Chapter 2 contains a review on Visual Servoing, including some basic definitions and formalizations, and the revision of some of the current hot research areas.

2. Chapter 3 contains the description of our approach to the Visual Servoing of Legged robots, namely the Aibo robot. Some of the main elements of this chapter are:

   (a) The description of how to obtain the direct kinematics of a legged robot, whose head has a camera.

   (b) The computation of the inverse kinematics to obtain the control commands.

   (c) The description of the Aibo realization of these ideas, including an actual implementation.

   (d) The description of the assessment experiment and its results.

3. Chapter 4 describes our works on Linked MCRS, from modeling to physical realization, Some outstanding sections are:

   (a) Building the hose geometrical model.

   (b) Adding the dynamics to the geometrical model.

   (c) Derivation of a control law for the Hose-MCRS system.

   (d) Simulation of the Hose-MCRS system.

   (e) Description of the real life system.

4. Appendix A contains additional information on modeling and interpolation using B-splines.

Chapter 2

Review in Visual Servoing

In this chapter we will perform a review of background ideas and the state of the art in Visual Servoing, aiming to provide a proper setting for the remaining of the dissertation. In section 2.1 we give some introductory comments. Section 2.2 gives some background ideas. Section 2.3 comments on the classification of the systems. In section 2.4 we give the formalization of the problem of visual servoing more broadly accepted nowadays. Finally, section 2.5 provides references on some active research areas, some of them of recent development.

2.1 Introduction

Visual Servoing is known as the task of positioning one or more robots in order to get poses of their final effectors using as feedback, in the closed control loop, the estimated positioning error computed from the visual information extracted of the environment by one or more video-cameras. For robotic manipulators the definition of Visual Servoing refers to the control of the pose of the final effector relative to a target defined by a set of image features. The pose of the final effector is defined as the position and orientation of the last element of its chain of articulations[1]. For mobile robots Visual Servoing refers to the robot pose relative to some landmarks detected in the environment. The systems that use a visual control in a closed loop do not need to know exactly the structure of the environment and the position of the robot articulations, because they can compensate the deviations trough the visual

[1]Across the dissertation we will use indistictly the terms "articulation" and "joint".

feedback. However, the visual feedback needs a high bandwidth and a high frequency in image processing.

The first works known in the Visual Servoing domain are from W. Wichman [83] in 1967 and Y. Shirai and H. Inoue [70] in 1973. The latter work coined the term "visual feedback" to identify the systems that use the visual information inside the closed control loop for robotic manipulators. At the end of the 70s some works about the use of visual control were developed in SRI International (originally known as Stanford Research Institute). Among the earliest works [63, 64] stand out, which describe the use of visual loops in two tasks: screwing and picking parts from a moving conveyor belt. The term Visual Servoing appeared for first time in the publication of J. Hill and W. T. Park [28] in 1979, where it was used for differentiating the real time visual control from the typical systems at that time which alternated between image capture and analysis and robot motion control. In 1981, Sanderson and Weiss [68] defined a classification of the Visual Servoing systems according to the space in which the error signal is defined, appearing for the first time the differentiation between Position Based Visual Servoing (PBVS) and Image Based Visual Servoing (IBVS). In the same work, they defined a classification between systems that use an internal control loop for the articulations (using positioning sensors, known as encoders), and those that instead of using encoders directly use the visual information in the control of the articulation positions.

In the 80s the development of this area was slow due to the difficulty in obaining computer hardware capable of doing the real time image processing at the high speeds that requires the servo-motor control. Before the appearance of personal computers the late of 80's the image processing specialized hardware was very expensive. In these early years some remarkable works appeared, such as [21] describing a screwing system using as feedback the information provided by a stereoscopic vision system. In 1984 Weiss [82] proposed an image feature based adaptive dynamic control of robots that tries to compensate the positioning error detected by the visual feedback information inside the closed control loop dynamics.

Also, in [28] a binary image processing for the estimation of positions and distances is described, based on distances between known features and showing that bi-dimensional and tri-dimensional visual information could be used for guiding robot movement for tracking and picking of moving parts. Similar works were developed in [55, 36] for the tracking of a swinging grasp, predicting the next position of the grasp. In [12] a video digital processing

system for determining the position of the target inside the image window is described, using the extracted information in a closed loop positioning control.

The technological advances at the beginning of the 90's allowed a greater spread of the approach, improving control results and leading to an increase in scientific publications. An exhaustive revision of the Visual Servoing with a huge bibliography was compiled by Corke [9] in 1993. It contains a description of the historical evolution and the main applications reported up to the publication time. In 1996, S. Hutchinson et al. [34] developed a tutorial on Visual Servoing which has been used as a reference since then and it is considered as the best beginners introductory material for this area, summarizing the diverse applications of Visual Servoing.

From the 90s the study of the relationship between robots and vision systems has acquired great importance as a research field. The main advantage of vision systems is that they allow to get an comprehensive description of the environment, with huge amounts of information, in a non intrusive way. They promise to be specially useful in low structured environments, where the environment features are constantly varying.

Generally, the industrial applications that use visual information for positioning the robot effector work in an open loop system, known as "Look-then-move", in which the visual information is analyzed in order to get an environment description and then plan the actions consequently; this model has some disadvantages, such as a high sensitivity to the environment perturbations and the precision of the system's calibration. Visual feedback allows to improve the systems performance and reduce its sensitivity (increase robustness) to perturbations and calibrations errors.

There is a lot of knowledge fields that participate in the design and building of Visual Servoing systems, namely the real time image analysis, the robot kinematics and dynamics modelling, control theory, real time computing, visual recognition, tracking or tri-dimensional recovery.

2.2 Background

2.2.1 Image features

In a control system design that uses visual sensorization, the first step is to determine which relevant information will be extracted from the camera

images. The kind of information may be very simple, such as points, or more complex, such as curves, surfaces, specific patterns or global properties of the image. The visual information analysis consumes a lot of computing resources depending on the image resolution. Some techniques initially work on a low resolution image, focusing inside interesting windows in the image which allow to analise with more detail the image areas where some features are expected to be found.

An *image feature* is defined as any structural information that can be extracted from the image. Every feature corresponds to the projection of a real physical feature on the camera plane. An *image feature parameter* is defined as any measurable real value that can be computed from one or more image features. Some examples of image feature parameters are the coordinates in the camera reference system of corners or the points of a line, the distance between two points or the orientation of the straight line that joints them. Another possibility may be the centroid of a point cloud or other more complex features.

From a set of k image feature parameters the *image feature parameter vector* is defined as $\mathbf{s} = [s_1, \ldots, s_k]^T$, with $\mathbf{s} \in \mathcal{F}$, being $\mathcal{F} \subset \mathbb{R}^{\mathrm{k}}$ the image feature parameter space.

From the position of a point in the task space[2], the position of its projection on the image plane is defined taking into account the *camera system model*. The traditional camera model used is the perspective projection, known as pin-hole camera, in which all the light rays that come from an object pass through a point known as projection center and fall on the image plane. To avoid inversion, sometimes the image plane is represented between the object and the projection center. In figure 2.1 we illustrate the projection of a point over the image plane.

The camera reference system is placed at the projection center, being the x-axis aligned with the optical axis. The image plane is parallel to the plane defined by the y-axis and z-axis, and it is placed at a distance λ over the x-axis from the origin of the camera reference system, known as *focal distance*. The intersection between the x-axis and the image plane is known as the *main point* and defines the origin of the image reference system, which has the u-axis and the v-axis parallel to the y-axis and the z-axis of the camera

[2]Task space is the physical space where the robot is placed, with some restrictions added due to the physical robot limitations that impede it reaching some regions of the space.

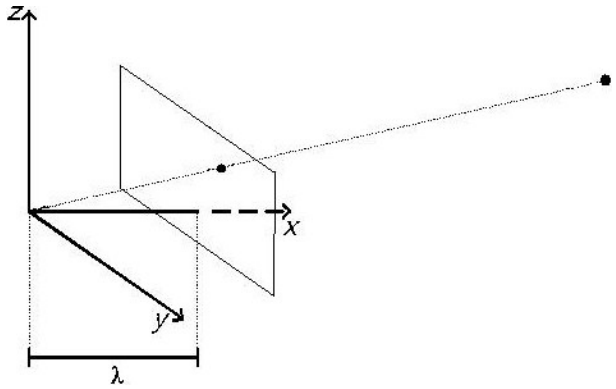


Figure 2.1: Camera reference system and camera projective transformation.

reference system, respectively.

The projection of a point $P = \{x, y, z\}$ expressed in the camera reference system has the following coordinates (u, v) in the image reference system:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \frac{\lambda}{x} \begin{pmatrix} y \\ z \end{pmatrix} \tag{2.1}$$

2.2.2 Camera configuration

There are two kinds of basic configurations for the camera according to [34], *eye-in-hand* and *fixed camera*.

In the eye-in-hand systems (figure 2.2), the camera is placed on the final robot's effector, allowing to perform the video acquisition inside the working space[3], centering the image capture process on the target object. In this system, the relation between the camera pose[4] and the robot's final effector pose is known and constant, because both share the same motion vector. The main disadvantage of this model is the possibility of loosing track of the target object as the robot movements may displace the target object outside the camera field of view.

On the other hand, in fixed camera systems (figure 2.3) the camera is placed in a fixed position inside the task space, separated from the robot, so it can capture images of the robot and the working space simultaneously.

[3]Working space is a region of task space where the robot's effector interacts with the objects it is working with.

[4]The pose is the physical position and orientation

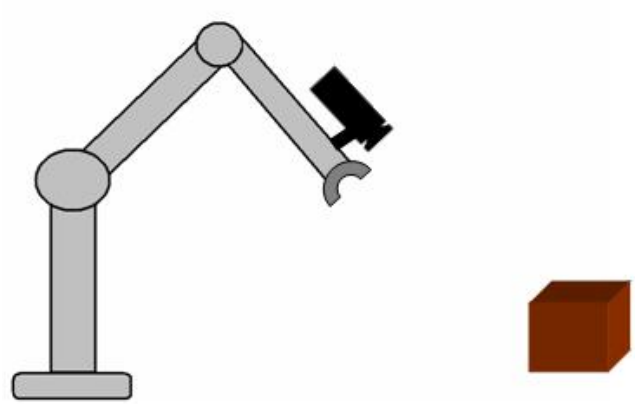


Figure 2.2: Eye-in-hand camera configuration.

In contrast with the previous case, in the fixed camera model the captured images are independent of the robot movements and there is a fixed relationship between the camera reference system and the task reference system which usually has its origin at the robot base.

For both models, eye-in-hand and fixed camera, some kind of camera calibration is needed. For the fixed camera it is necessary to calibrate the camera intrinsic (internal geometry and optic features) and extrinsic parameters (position and orientation). For the eye-in-hand, the camera intrinsic paremeters may be required.

Frequently, the robotic tasks are defined relative to one or more coordinate reference systems. For example, the pose of an object as obtained from the image processing is expressed in the camera reference system, while the pose of a target to be picked by a robot is usually expressed in the task reference system. Given two reference systems, it is possible to express the relationship between them by the composition of the transformations from one reference system into the other. These transformations usually are rotations and translations. If we denote ${}_bI_c$ the matrix transformation from the camera reference system into the robot base reference system, and we denote ${}_eI_b$ the transformation from the robot base reference system into the robot final effector reference system, then a point P in the camera reference system can be expressed in the robot final effector reference system by the following expression: $({}_eI_b \cdot {}_bI_c)P$.

When performing tri-dimensional reconstruction from the image features it is necessary to have additional information (e.g. stereo vision) that allows

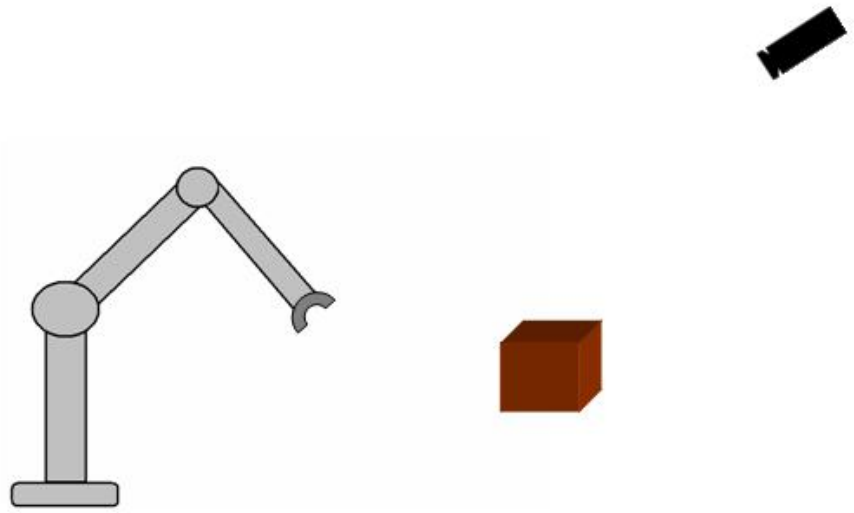


Figure 2.3: Fixed camera configuration.

to determine the 3D space coordinates of the physical object features. Generally it is used to determine the feature point's depth (its x-axis coordinate in the camera reference system of figure 2.1 or, in other words, its distance to the image plane) so we can compute its 3D coordinates in the camera space by the following equation

$$(x, y, z)^T = (p, \frac{pu}{\lambda}, \frac{pv}{\lambda}),$$

where p is the point's depth and λ the camera focal distance. In order to detect the depth of a point using only one static camera it is necessary to obtain this additional information from the redundancy of the image features. Asuming the target does not move significantly from one view to the other, sequences of views from one moving camera may be processed to derive depth information. However, this kind of approaches have the disadvantage of the ocurrence of singularities and local minima. Some works use stereo vision to get a tri-dimensional reconstruction using epipolar geometry properties. It is assumed that the first Visual Servoing work using stereo vision was the screwing system of [21]. Other pioneering system using two video cameras for Visual Servoing is [42], which used a pair of parallel video cameras to estimate the image features' Jacobian. In [33] a trajectory generator uses a stereo vision system to avoid obstacles. They report that the stereo system

added robustness and smoothness to the movements of the robot.

Some works use more than two cameras, which incorporate redundant information that give certain robustness against partial occlusions at the expense of increasing the execution time. In [37] it is defined a multi-camera Visual Servoing system, showing experimental results using image based Visual Servoing (2D) and hybrid Visual Servoing ($2\frac{1}{2}$D) with two cameras that observe two different views of an object. In [41] it is assumed that there is no *a priori* knowledge of the object model, so initially the robot's effector is positioned with respect to the target performing learning movements around it, using the obtained information about the relationship between the robot final effector and the camera is used to compose the Jacobian matrices of both cameras. In [5] the authors report empirical results that show a greater convergence to a linear trajectory of the robot when using 3D coordinates instead of 2D coordinates, and as consequence the image features tend to keep inside of the camera range in the 3D coordinate based systems when performing the required task movements .

2.3 Architectures and classifications

In the seminal review work by Sanderson and Weiss [68] in 1980, the Visual Servoing systems are classified according to the space in which the error signal is defined. Other classification is built, according to the kind of feedback at the joint level between systems which use internal control loops from position sensors (encoders) and those which directly use the visual information in the computation of the goal joint positions.

2.3.1 Classification according to the joint feedback

Sanderson and Weiss [68] classified the Visual Servoing systems into *look-then-move* and *visual servo control*. Since the term visual servo control finally became a standard to generically describe any kind of visual control for a robotic system, it has been taken the convention of defining the look-then-move systems as *indirect Visual Servoing* systems and the systems that Sanderson and Weiss called visual servo control systems as *direct Visual Servoing*.

In the indirect Visual Servoing systems (figure 2.4) it exists a controller at the level of the articulations of the robot, with a feedback loop that performs

the local control of the articulation based on encoder information to reach the goal positions set by the Visual Servoing loop of the system. Two main classes of indirect systems exist; the first one receives the name *static look-then-move* and works in a sequential way, capturing an image, processing it and then sending the motion command to the robot joint controllers. The robot, then, executes a movement assuming that the environment remains invariant. The Visual Servoing sub-system waits until the robot joints reach the desired position to start a new cycle. This control scheme maintains separated the joint and visual control loops. The second class receives the name *dynamic look-then-move*. These systems do not wait until the desired positions of the joints have been reached to start processing a new image: the joint and visual control loops are interleaved because the visual control loop allows to update the desired positions of the joints while the robot still continues executing the previous movement; in these systems the joint loop runs at a greater frequency than the visual loop.

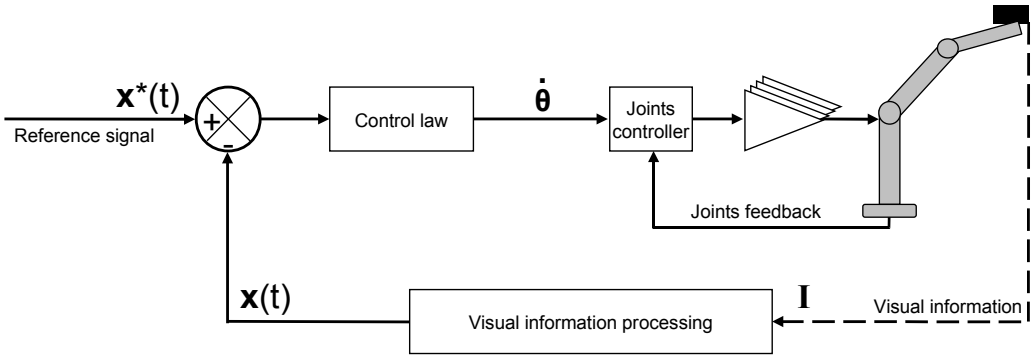


Figure 2.4: Indirect Visual Servoing systems (look-then-move)

In direct Visual Servoing systems (figure 2.5) there is not a control loop at the joint level; the robot joint positions are computed from the visual information at the sampling frequency of the camera. In this case the visual control loop performs the control and stabilization of the servomotors of the robot. This approach was less used in early systems because of:

- The high frequency required of the visual information processing sub-system,
- The fact that most of the robot's servomotors have incorporated an interface that allows incremental commands in Cartesian position and

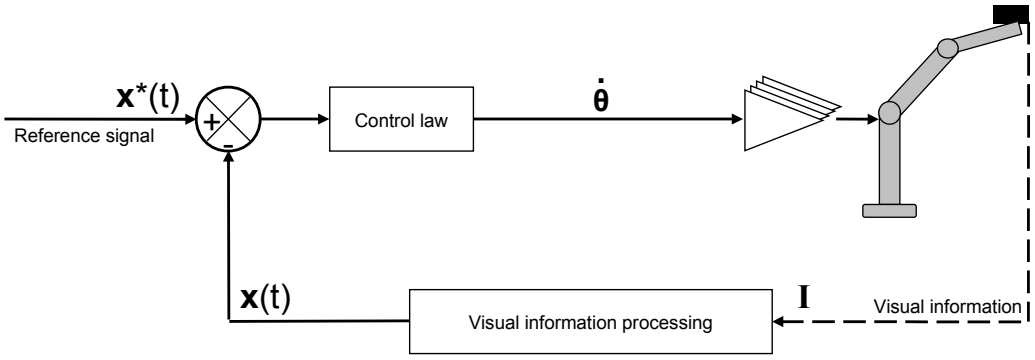


Figure 2.5: Direct Visual Servoing systems.

velocity, which simplifies the construction of indirect Visual Servoing systems.

- The excessive influence of perturbations in the real time estimation of the current joint states from encoder information.

2.3.2 Classification according to the control space

The classification of Visual Servoing systems according on the control space distinguishes between *position based control* and *image based control*.

Position Based Visual Servoing

In Position Based Visual Servoing (PBVS), illustrated in figure 2.6, it is assumed that an *a priori* knowledge of the environment's structure, the target object and the camera is given. The image features, denoted as $\mathbf{s}(\mathbf{t})$, are extracted and used to estimate the pose of the target object $\mathbf{x}(t)$ relative to the task reference system through a tri-dimensional reconstruction of the environment's structure. The difference between the desired and the current pose of the target object constitutes the input error signal to the control system. In these systems, there is a separation between the control needed to perform the assigned task and the computational process of estimating the target pose relative to the task reference system; so, the error signal is defined in the task space or, in other words, in the 3D space.

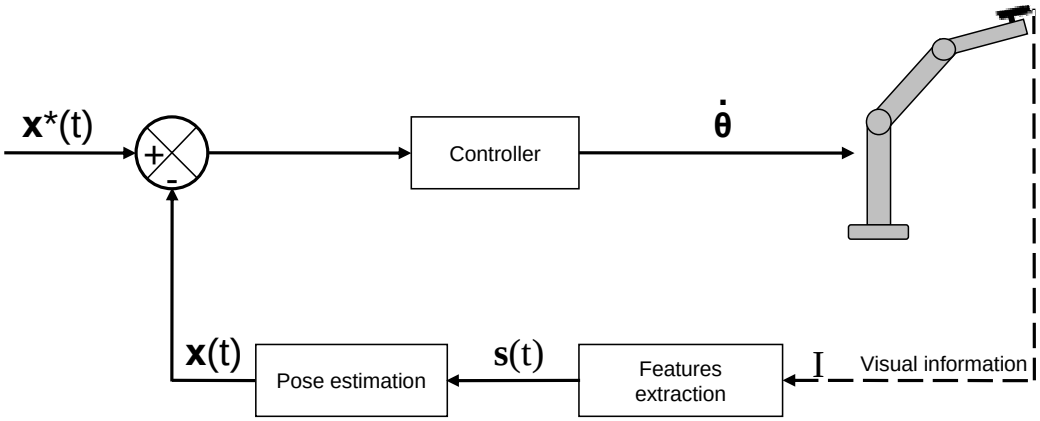


Figure 2.6: PBVS - Position Based Visual Servoing.

Image Based Visual Servoing

In Image Based Visual Servoing (IBVS), illustrated in figure 2.7, the values of the control parameters are computed as a direct function of the image features. Unlike PBVS, the error signal is defined in the bi-dimensional image reference system and it is used directly as the input to the control system. For this reason, the IBVS is also known as 2D Visual Servoing system. In this case there is a direct influence of the image features on the state of the robot joints. This relationship is encapsulated in the image Jacobian matrix, because it defines a transformation between the variations in the pose of the target object in the camera reference system and the observed image feature variations.

The most common approach to bring the image feature parameter values to the desired ones is to perform a local minimization of the error through a local linearization of the robot kinematic function. That is, we compute a linear relation between variations in image features and variations in the robot pose. In general, an *a priori* knowledge of the image geometric features is needed, such as corners or edges [7, 11, 23] or visual landmarks [16]. When there is no knowledge about geometric features, an alternative is the image motion-based Visual Servoing [58, 75, 13, 14] (also known as 2D+dt Visual Servoing), which uses the estimated motion fields of the perceived motion in the image plane between two successive images as feedback into the control loop.

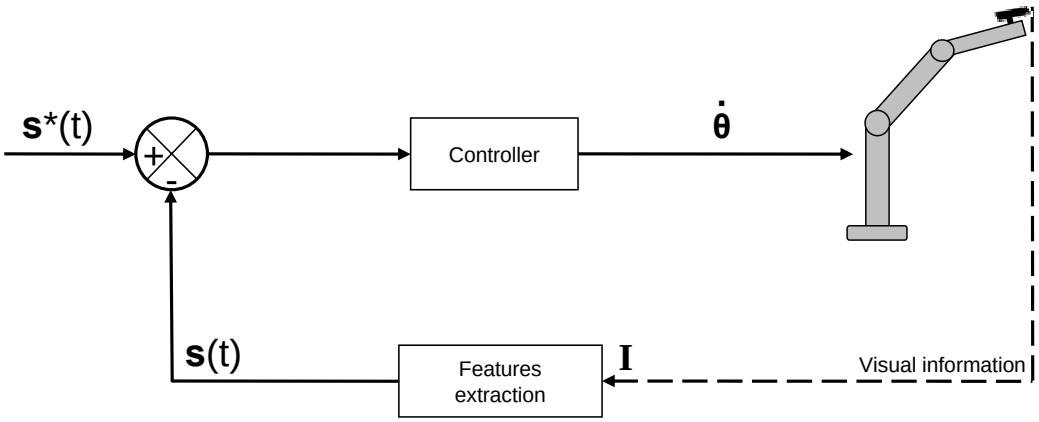


Figure 2.7: IBVS - Image Based Visual Servoing.

Comparison between IBVS and PBVS

The basic difference between the IBVS and PBVS approaches lies in the definition of the error signal space. PBVS is more sensitive to calibration errors, because they introduce perturbations and deviations in the tri-dimensional reconstruction of the environment, generating errors in the execution of the robot movements, whereas IBVS systems link directly the image features to the robot joints.

In PBVS, due to the fact that the task is defined based on the localization of the robot and the target object in the task space, we can compute the desired robot trajectory following a straight line, so there are no local minima of the error in the minimization process.

The main advantage of IBVS systems is the fact that precision is independent of the calibration and the exact knowledge of the target object model. In contrast, it is not possible to assure global stability and usually singularities in the transformations between image features and robot joints appear as, for example, in case of image feature occlusions. Another example is that some movements do not induce changes in image features, so they introduce singularities in the Jacobian matrix. Because of that, finding an inverse of the Jacobian matrix turns out to be difficult if it is not enough well conditioned. The desired trajectories in the image plane can be computed as straight lines, but its transformation into the joint space using the local image Jacobian matrices may generate impossible joint trajectories.

In 1997 Malis et al. [6] published a new approach which is a middle point

Symbol	Description
e	Kynematic error.
$\mathbf{x}$	State vector.
$\boldsymbol{v}_r$	Robot's final effector velocity.
$\boldsymbol{\theta}$	Robot's joint vector.
$\mathbf{C}$	Combination matrix.
J_e	Task Jacobian.
L_e	Error interaction matrix.
J_r	Robot Jacobian.
L_x	State interaction matrix.
$\mathbf{r}$	Final effector pose.
λ	Convergence rate.
L_s	Image interaction matrix.

Table 2.1: Notation for the control specification

between both classic systems, that makes use of the advantages and avoids the disadvantages of PBVS and IBVS. It incorporates the advantage of not needing a geometrical model of the target from IBVS, and the possibility of assuring the convergence of the control law in the work space from the PBVS. This new approach uses 3D information and 2D information, so it is denominated as Visual Servoing $2\frac{1}{2}$ or *Hybrid Visual Servoing*.

2.4 System Control formalization

In this section we analize the control system, that is, the computational problem of obtaining the robot's final effector velocity $\mathbf{v}_r \in \mathbb{R}^6$ in the task Cartesian space as a function of the error of the state variable $\mathbf{x}$. The positioning task ends when $\mathbf{e}(\boldsymbol{\theta}, t^*) = 0$, where t^* is the instant in which the state reaches the value $\mathbf{x}^*$, and $\boldsymbol{\theta}$ is the vector of joint angles of the whole robot.

Positioning kinematic error

In [62, 67] the robotic problem is defined as a reduction to zero of the positioning error $\|\mathbf{x}(\boldsymbol{\theta}, t) - \mathbf{x}^*\|$ in a finite time. The *Kinematic error function,*

$e : \mathbb{R}^m \rightarrow \mathbb{R}^n$, is defined as:

$$e(\boldsymbol{\theta}, t) = \mathbf{C} \cdot (\mathbf{x}(\boldsymbol{\theta}, t) - \mathbf{x}^*), \tag{2.2}$$

where:

- $\boldsymbol{\theta} \in \boldsymbol{T}$ is the vector of the robot joint positions,
- $\mathbf{x}(\boldsymbol{\theta}, t)$ is the vector of the configuration computed from visual information, that is, the pose of the final effector for PBVS and the image features for IBVS.
- $\mathbf{x}^*$ the desired visual configuration.
- $\mathbf{C}$ the *combination matrix* of dimension $n \times m$, being n the number of degrees of freedom and m the dimension of the state vector.

It is also defined the *Task Jacobian* J_e as:

$$J_e = \frac{\partial e}{\partial \boldsymbol{\theta}}, \tag{2.3}$$

which is usually represented as the composition of two Jacobian matrices:

$$J_e = L_e \cdot J_r, \tag{2.4}$$

where:

- $L_e = \frac{\partial e}{\partial \mathbf{r}} \in \mathbb{R}^{n \times 6}$, is the *Error Interaction matrix*, the matrix that relates the kinematic error with the robot Cartesian velocity.
- $J_r = \frac{\partial \mathbf{r}}{\partial \theta} \in \mathbb{R}^{6 \times n}$ is the *Robot Jacobian*, the matrix that relates the velocity of the final effector in the Cartesian space with the velocities at the joints of the robot.

It is said that the task is *admissible* if there is an unique trajectory of $\boldsymbol{\theta}$ for which the error function reaches zero at the time limit ($e(\boldsymbol{\theta}, t^*) = \mathbf{0}$) and at the same time J_e is regular all over this trajectory. For the IBVS systems, admissibility requires the visibility condition, i. e. that there are enough visual features inside the vision range of the camera, while for the PBVS systems and hybrid systems, besides the visibility property, it is also required the estimation of the target object pose respect to the camera.

Final effector trajectory in the task space

The aim of the Visual Servoing system is the positioning of the final effector of the robot respect to a target object, so it is assumed that the vector state $\mathbf{x}$ is differentiable as a function of the robot's final effector pose $\mathbf{r}$. The velocity of the state vector can be expressed as a function of the velocity of the robot's final effector pose respect to the target object:

$$\dot{\mathbf{x}} = \frac{\partial \mathbf{x}}{\partial \mathbf{r}} \frac{\partial \mathbf{r}}{\partial t} + \frac{\partial \mathbf{x}}{\partial t}. \tag{2.5}$$

Defining the *State Interaction matrix* $L_x = \frac{\partial \mathbf{x}}{\partial \mathbf{r}}$ as the Jacobian matrix that relates the state velocity with the velocity of the robot's final effector in the camera space $\mathbf{v}_r = \frac{\partial \mathbf{r}}{\partial t}$, equation 2.5 can be rewritten in the following way:

$$\dot{\mathbf{x}} = L_x \cdot \mathbf{v}_r + \frac{\partial \mathbf{x}}{\partial t}. \tag{2.6}$$

At the same time, we can use the error interaction matrix L_e, which models the sensitivity of the task error respect to the velocity of the robot's final effector. If we suppose that the combination matrix $\mathbf{C}$ does not depend explicitly on $\mathbf{r}$ in its definition, the error interaction matrix can be rewritten as:

$$L_e = \mathbf{C} L_x. \tag{2.7}$$

If we want an exponential decrease of the kinematic error (equation 2.2) in time, using the time constant λ the error trajectory can be described by the following linear differential equation:

$$\dot{e} = -\lambda e. \tag{2.8}$$

The time derivate of the error kinematic function, $\dot{e}$, can be written as a function of the velocity of the robot's joints, $\dot{\theta}$, and, if we also use the decomposition $\frac{\partial e}{\partial \boldsymbol{\theta}} = \frac{\partial e}{\partial \mathbf{r}} \frac{\partial \mathbf{r}}{\partial \boldsymbol{\theta}}$, we get the following expression:

$$\dot{e} = \frac{\partial e}{\partial \mathbf{r}} \frac{\partial \mathbf{r}}{\partial \boldsymbol{\theta}} \dot{\boldsymbol{\theta}} + \frac{\partial e}{\partial t}. \tag{2.9}$$

The expression $\frac{\partial e}{\partial \mathbf{r}}$ is the error interaction matrix L_e (see equation 2.3), while the expression $\frac{\partial \mathbf{r}}{\partial \boldsymbol{\theta}} \dot{\boldsymbol{\theta}}$ can be substituted by the velocity of the robot's

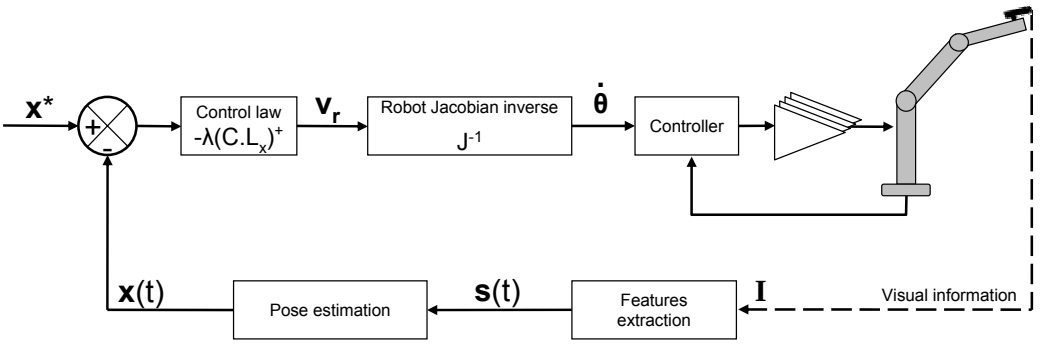


Figure 2.8: General schema for Visual Servoing control.

final effector $\mathbf{v}_r$. So, we get the following simplified expression of equation 2.9:

$$\dot{e} = L_e.\mathbf{v}_r + \frac{\partial e}{\partial t}. \tag{2.10}$$

Isolating $\mathbf{v}_r$ from equation 2.10 we obtain the following expression:

$$\mathbf{v}_r = L_e^+ \left(\dot{e} - \frac{\partial e}{\partial t} \right), \tag{2.11}$$

where L_e^+ is the pseudo inverse of the error interaction matrix.

Substituting $\dot{e}$ in equation 2.11 by its expression in equation 2.8 we get the following expression for the robot's final effector velocity as a function of the kinematic error:

$$\mathbf{v}_r = -L_e^+ \left(\lambda e + \frac{\partial e}{\partial t} \right) \tag{2.12}$$

However L_e and $\frac{\partial e}{\partial t}$ can only be estimated from the visual information, so the Cartesian velocity of the robot's final effector is formally defined as:

$$\widehat{\mathbf{v}_r} = -\widehat{L_e}^+ \left(\lambda e + \widehat{\frac{\partial e}{\partial t}} \right) \tag{2.13}$$

Stability and convergence Substituting $\widehat{\mathbf{v}_\mathbf{r}}$ by expression 2.13 in equation 2.10, we get the following expression for the velocity of the kinematic error function:

$$\dot{e} = -\widehat{L_e L_e}^{+} \left(\lambda e + \widehat{\frac{\partial e}{\partial t}} \right) + \frac{\partial e}{\partial t}. \tag{2.14}$$

The following sufficient condition assures the steady decrease of the norm of the kinematic error function $\|e\|$:

$$\widehat{L_e L_e}^{+} > 0. \tag{2.15}$$

Assuming that the combination matrix $\mathbf{C}$ does not contain explicitly $\mathbf{r}$ in its definition, we can rewrite the previous condition as:

$$\mathbf{C} L_x (\mathbf{C} \hat{L_x})^{+} > 0. \tag{2.16}$$

The state vector $\mathbf{x}$ represents the information used as feedback in the control loop. Depending on the architecture of the system it can represents the image feature parameter vector on IBVS or the pose of the robot's final effector on PBVS.

When dealing with IBVS, the standard notation is $\mathbf{s}$ instead of $\mathbf{x}$. The error interaction matrix is then expressed as:

$$\hat{L_e} = \mathbf{C} \hat{L_s}, \tag{2.17}$$

being $L_s = \frac{\partial \mathbf{s}}{\partial \mathbf{r}}$ the Jacobian of the image feature parameter vector as a function of the pose of the robot's final effector. Generally the interaction matrix is chosen equal to the identity matrix, so the stability condition reduces to:

$$L_s \hat{L_s}^{+} > 0. \tag{2.18}$$

When the control is PBVS, the state interaction matrix is expressed as the following matrix composition:

$$L_x = L_{xs} L_s, \tag{2.19}$$

where matrix L_{xs} represents the 3D reconstruction from the image features.

The stability condition is defined in the following way:

$$L_x \hat{L_x}^{+} > 0. \tag{2.20}$$

2.5 Active research areas

In this section we give a few glimpses abount the current focus of interest in the development of Visual Servoing systems. It may be of use to set a framework for our own works, although such a categorization may not always be easy.

2.5.1 Trajectory generation

In PBVS it is easy to generate the robot's reference trajectory due to the fact that the target information is given in 3D. In contrast, in IBVS the visual information is produced in the 2D image plane, and therefore some of the intermediate positions in task space corresponding to ones in the image space trajectory defined by the vision based control may not be reachable by the robot. As a consecuence, it is recommended that the desired positions of the image features should not be far from their current position to minimize the risk of planning unfeasible trajectories.

Despite this disadvantages, IBVS may be preferred in many real life situations because it is less sensitive to vision calibration errors than PBVS (i.e. more robuts to imprecise positioning). When distance between initial and final positions in the image is big, we want to optimize the trajectory in the working space minimizing the image error function, with the additional objective of keeping the features inside the image all the time while following the trajectory. The task of finding an optimum trajectory that allows the target object to follow a desired path in the image is known as *trajectory generation.*

The first work on trajectory generation, to our best knowledge, was developed by J. Feddema in 1989 [17] mantaining a continous trajectory by matching the velocity, accelerations and accelerations derivatives of the features but not their position. In [66] an on-line trajectory generation method is presented where a weighting matrix is chosen taking into account a maximum velocity for each joint and applying the weighted least norm method. This matrix has the effect of suppressing the reference velocity. A new approach considering the constraint of maintaining the target object in the camera field of view is proposed in [43], where a potential field that induces repulsive forces is defined to create a potential barrier around the camera field of view in order to assure that all the features are always observable. In [51] a trajectory of a gripper which moves over a straight line is generated

using an uncalibrated stereo rig; the trajectory is computed by decomposing the projective coordinates of initial and desired points into a special rigid displacement and a triangular matrix. A similar work is developed in [53, 52] where intermediate configurations between initial points an desired ones are constructed in the projective space, using a conjugate transformation and projective invariants, and then reprojected onto the image planes to get an image-based trajectory. In [44] a modified potential field method is used to determine discrete trayectories that are then interpolated by b-splines in order to obtain continous curves that introduce an improvement of the dynamic behaviour of the system.

2.5.2 Integration of Visual Servoing and force control

The use of combinations of Visual Servoing and force control has been growing since the increasing processing power and lowering cost of vision systems have allowed the generalization of the aplicability of this approach. This combination is highly complementary because force sensors contribute 3D information about the contact between the robot and the target object, while vision sensors give information about the 3D environment. The devolopment of force and vision sensor integration has been based on the area of force controlled manipulators.

In force controlled manipulators [69] we find two main approaches: hybrid position/force control [60] and impendance control [29]. In hybrid position/force control, the control space is separated into position and force regions, defining a feedback loop for force control and another feedback loop for position control, which work independently and in parallel. The force subspace is known as wrench space while the position subspace is known as twist space. Hybrid approaches allow faster dynamics but require model-based compensation. In impedance control a relationship between motion and force is established by translating a task into a desired impedance.

The main problem for integrating vision and force information is that they do not share a common data representation and, in consecuence, need to be used in differente stages of the control system. An impedance based visual/force approach was defined in [48], where a level decomposition view of the use of vision/force controller is established defining three independent tasks of traded, hybrid and shared control.

In [30] an adaptative hybrid visual/force controller is used to do Visual Servoing while the robot applies contact forces on a surface, which has an

on-line estimator for the parameters of the unknown constraint surface that only needs the knowledge about the manipulator kinematics. A control algorithm in the impedance control approach is defined in [47], performing a peg in a hole insertion using a 7 axis robot manipulator where the reference trayectory to the impedance controller is generated on-line by an IBVS loop. An hybrid control approach is presented in [56], where an extraction of a book on a shelf is developed. In [69] a framework based on the Task Frame Formalism (TFF) for distinguishing between different types of shared control is presented. A method for tracking trajectories, known as movement flow-based Visual Servoing, is presented in [54]; this method uses the Kalman filter as the criteria for asigning weights to variables for each sensor system.

2.5.3 Invariant Visual Servoing

In Visual servoing the task can be clasiffied by the knowledge or ignorance of the target object model. If a model of the target object is known a model-based aproach may be used. In contrast, if there is not a model of the target a model-free approach is used. In the model-free approach an initial learning step is performed in order to get reference images of the target that allow to estimate the parameters of an object model. Changes in the intrinsic parameters of the camera affect the servoing performance, so a new learning step should be neccesary.

The Invariant Visual Servoing aproach was introduced by Ezio Mails [38] and an extension of this work is followed by [40], trying to extend the teaching-by-showing technique when different cameras are used for teaching and servoing, without an explicit calibration between them. This new approach works in a projective space invariant to camera intrinsic parameters and to the knowledge of the tri-dimensional model of the target object, and allows the use of differente cameras for the learning step and servoing task. In [18] a redefinition of invariant Visual Servoing approach is developed in order to allow the use of zooming during a positioning task, using weighted image features to avoid the discontinuities produced by the appeareance and disappearance of image features during the control task. In [19] a study on how to select some of the parameters of the weight function is done; a stability analysis of invariant Visual Servoing with weighted features is also proposed. In [39] the use of the invariant Visual Servoing approach is proposed for the reconstruction of underwater objects, simulating an active underwater stereovision system mounted on a 6 DOF manipulator arm effector over an

underwater vehicle.

2.5.4 Partitioned Visual Servoing

In IBVS the image Jacobian defines a mapping between image space velocities and velocities of the robot joints, it results in an advantage with respect to position based approaches since it does not need a tri-dimensional reconstruction. But on the other hand, the pure use of the image Jacobian lead to control problems because it is poor conditioned and may be prone to the ocurrence of singularities. Moreover, because this approach works on the image plane the trayectories of the robot in the cartesian space are quite folded and may drive the robot towards singularities in the image Jacobian.

In [8] a partitioning approach is introduced trying to avoid this problem by decoupling the motions in the axis of the camera reference system perpendiculars to the image plane, usually known as z-axis. The traditional IBVS control takes the form

$$\dot{r}_{xyz} = J^{+}_{xyz}\dot{\mathbf{s}},$$

while under this partitioning approach the control takes the form

$$\dot{\mathbf{r}}_{xy} = J^{+}_{xy}\{\mathbf{s} - \dot{\mathbf{J}}_{\mathbf{z}}\dot{\mathbf{r}}_{\mathbf{z}}\},$$

for the xy movements, being $\mathbf{s}$ the feature point coordinate error and J_{xyz}, J_{xy}, J_z the respective images Jacobian for the three cartesian axes of the camera reference system, the camera plane axes (x, y) and the z-axis perpendicular to the image plane. The movements in the z-axis are described based on two new features for translational and rotational movements.

2.5.5 Neural Networks

In IBVS the image Jacobian is needed in order to define a relation between the image feature space and the robot's movements space. This matrix is not easily constructed even with full knowledge of the robot's kinematics and the camera model. Moreover, the use of the inverse of the image Jacobian does not serve well to compute large feature movements in the image due to the large motion errors derived from the implicit linearization and, in the worst case, the image Jacobian might be singular. Some approaches have tried to avoid the inconveniences derived from the image Jacobian using a trajectory generation.

The works done in [45, 46] proposed the use of neural networks in the learning of a control system for Visual Servoing, where the *a priori* knowledge of the robot kinematics or the pose of the target respect to the robot was not assumed. Then, in [25] a self-organizing Visual Servoing system was proposed, where the learning of the image Jacobian is done despite the geometric dimensions. The use of a fuzzy controller with a supervised capability was proposed in [74], where the elements of the image Jacobian do not take into account the relative distance between the target and the robot, using only the informaiton about the image features, however a satisfactory performance could not be assured due tho the simple gradient method used by the learning algorithm. Later, in [72] this approach was improved by defining a fuzzy membership function based neural network (FMFNN) for approximating the nonlinear mapping avoiding the use of the inverse of the Jacobian. This approach trains the network to generate fast movements in the robot when the target is far away and more slow movements when it is near the desired features, although this approach still does not make use of the robot dynamics in determining the desired feature trajectories, and only the simulation results were presented. In order to take into account the robot dynamics, the work in [73] was done presenting the results with a real robot, where the robot first moves in a perpendicular direction to the object and then an orientational motion control is applied only when the target object is near its desired pose. Some works [4, 78, 77, 27] define the neuro-controller as a composition of subnetwork modules.

In [81] a Neural Network approach to multisensory Visual Servoing is presented, where a multilayer perceptron network is used to learn the direct mapping from multisensory data to robot motions. The main advantage is that the goal position can be changed without having to perform network retraining. In [20] a controller based on learned behaviour by trial and error without needing calibration is presented, where remarks the use of continous states and actions. In [35] the study of the stability in the use of Neural Networks compensation for closed-loop system with 2D visual information is developed. In [85] a Sliding-mode observer is used in order to estimate de velocities of joints, despite using joint velocity sensors, and a RBF Neural Network in order to compensate gravity and friction. In [86] RBF Neural Networks are used in order to compensate the uncertainties associated with robot dynamics and the Jacobian matrix. In [71] the Evolutionary Acquisition of Neural Topologies (EANT) is presented, which is a method to learn Neural Networks from a minimal neural structure which grows using evolu-

tionary reinforcement learning. In [24] a Visual Servoing system that uses Takagi-Sugeno Fuzzy Neural Network controller is developed, where neither artificial marks, a priori knowledge of the robot kinetics, dynamics nor camera calibration are required. In [50] a combination of a PI kinematic controller and a Feedforward Neural Network (FFNN) is used in order to achieve a desired tracking; the controller is responsible for achieving the motion of the target in the the image plane, while the FFNN is responsabile for computing the required torques to compensate the robot dynamics. In [2] an EMRAN-RBF neural network is used for estimating the Jacobian, which allows the mimetic control of a robot with different dynamics.

Chapter 3

Visual Servoing of Legged Robots

This chapter presents a contribution to the visual tracking of objects by a legged robot using all of its degrees of freedom. We approach this issue in a principled way applying ideas of Visual Servoing. Nowadays visual tracking solutions for this kind of robots inspired in the Visual Servoing approach only move the effectors linked directly to the camera. As far as we know, not much work has been reported in the literature about Visual Servoing for legged robots, giving general descriptions of the control system [31, 32]. In this work we concentrate in obtaining a detailed mathematical description of the image Jacobian matrix taking into account all the effectors which can affect the image captured by the robot's camera. In section 3.1 we provide a general description of the approach. In section 3.2 we construct, starting from a general description of this kind of robots, the Jacobian matrix that describes its forward kinematics. Visual Servoing is performed computing the pseudoinverse of this matrix in section 3.3. In section 3.4 we specialize our approach for an Aibo robot, presenting some experimental empirical results on the actual performance of the approach and discussing its limitations. The notation used through the chapter is presented in table 3.1.

3.1 General description of the approach

Our general approach to the Visual Servoing of legged robots follows the conventional lines of the Image Based Visual Servoing (IBVS) systems described in the previous chapter. We build a locally linear kinematic model of the robot by composing the diverse Jacobian matrices that embody the

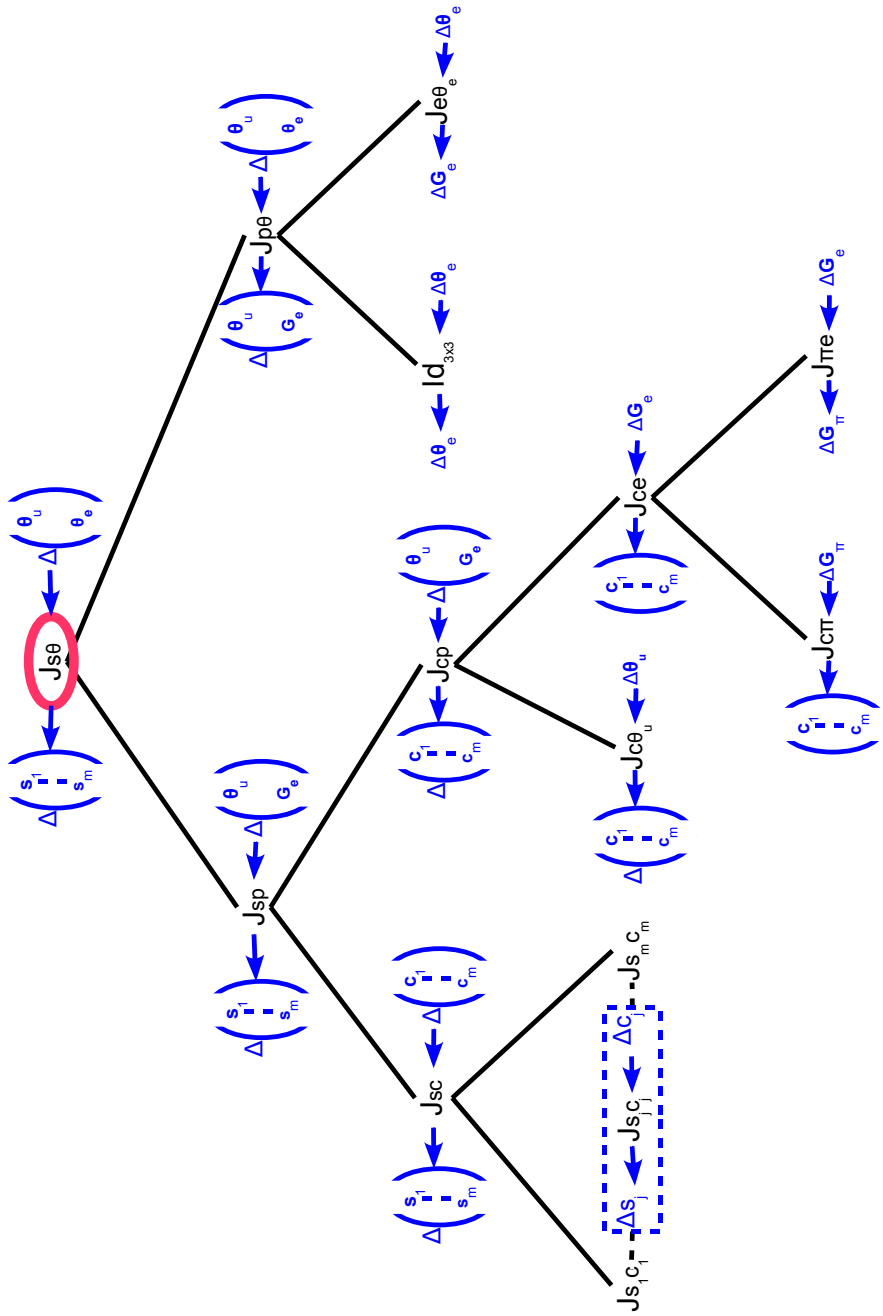


Figure 3.1: General structure of the operators composing the direct kinematics model.

I_c, I_b, I_g	Coordinate Reference Systems for the camera, body and ground.
${}^g\mathbf{s}, {}^c\mathbf{s}$	Feature point $\mathbf{s}$ expressed in the Coordinate Reference System I_g, I_c.
${}_iI_j$	Transformation between Coordinate Reference Systems.
θ_l, θ_u	Joint angles of the leg and upper body articulations.
$\mathbf{G}_\pi, \mathbf{G}_e$	Basic and extended sets of support points.
$\mathbf{p}$	The extended support points and the upper articulations.
$J_{s\theta}$	Dependence of image features on the robot articulations.
J_{sp}	Dependence of image features on $\mathbf{p}$.
J_{sc}	Dependence of image features on the camera.
$J_{p\theta}$	Dependence of $\mathbf{p}$ on the robot articulations.
$J_{c\theta_u}$	Dependence of the camera on the upper articulations.
J_{ce}	Dependence of the camera on $\mathbf{G}_e$.
$J_{c\pi}$	Dependence of the camera on $\mathbf{G}_\pi$.
$J_{\pi e}$	Dependence of $\mathbf{G}_\pi$ on $\mathbf{G}_e$.
$J_{e\theta}$	Dependence of $\mathbf{G}_e$on the robot articulations.

Table 3.1: Nomenclature used across the chapter.

dependences among observation and control parameters. Then we propose a simple inversion of the model to obtain the desired control commands that will accomplish the minimization of the perceptual error detected in the vision system. One of the key problems in the development of this approach is the definition of a ground reference system needed to relate the effect of the articulations on the perception of the objects in the world. This ground reference system is trivial for static manipulator robots, while it can be arbitrary and changing for legged and mobile robots. We have solved this problem by using the tips of the legs that are the ground contact points to define this ground reference system. Another basic problem was the determination of the ground plane upon which the robot is resting. We have solved it by using the joint state information provided by the robot's basic control systems.

To build the transformations between reference systems, we have reasoned as follows:

- From the joints' information obtained in body centered coordinates, we have defined the ground reference system.

- From the ground reference system we can formulate the dependence of the camera and body poses on the variations of the legs' articulations

and the upper body articulations.

In figure 3.1 we show as a tree structure the decomposition of the Jacobian matrices which give the linear model of the system kinematics of a legged robot. For each matrix we specify in blue characters the input and output system variables.

The system kinematics is described in full by the Jacobian matrix $J_{s\theta}$ which embodies the dependence of the image features on the robot articulations; it is decomposed as $J_{s\theta} = J_{sp}J_{p\theta}$. The Jacobian matrix J_{sp} embodies the dependence of the image features on the upper body articulations and the extended support points. The Jacobian matrix $J_{p\theta}$ embodies the dependence of the extended support point positions and upper body articulations on the robot articulations.

The Jacobian matrix J_{sp} is further decomposed as $J_{sp} = J_{sc}J_{cp}$, where J_{sc} embodies the dependence of the image features on the camera reference system and J_{cp} embodies the dependence of the camera reference system on the upper body degrees of freedom and the extended support points.

The Jacobian J_{sc} is constructed by aggregating the Jacobian matrices corresponding to each feature point into a block diagonal matrix. The Jacobian matrix J_{cp} is also a diagonal aggregation of matrices $J_{c\theta_u}$, that embodies the dependence of the camera reference system on the upper articulations, and J_{ce}, that embodies the dependence of the camera reference system on the extended support point positions.

The Jacobian matrix J_{ce} is further decomposed as $J_{ce} = J_{c\pi}J_{\pi e}$, where $J_{c\pi}$ embodies the dependence of the camera reference system on the basic support point positions, and $J_{\pi e}$ embodies the dependence of the basic support point positions on the extended support point positions.

Finally, the Jacobian $J_{p\theta}$ is a diagonal aggregation of the identity matrix $\mathbf{I}_{3\times3}$ and the Jacobian matrix $J_{e\theta}$, that embodies the dependence of the extended support point positions on the robot articulations.

3.2 Direct kinematics

We build the robot kinematics as a transformation from the ground supporting plane to the camera coordinate system, composing the diverse transformations that correspond to the limbs, and then going up trough the upper robot's degrees of freedom making the articulation chain from the body center to the camera.

As illustrated in the legged robot schematics in figure 3.2, the legs are the elements which support the robot's body and, therefore, define a relationship between the body and the support plane. We need, then, to be able to determine the 3D coordinates of the leg's points in contact with the ground at any time. We denote $\mathbf{g}_i$ the support points which are the legs' tip points in contact with the ground. The support points are highlighted by a red circle around them in figure 3.2.

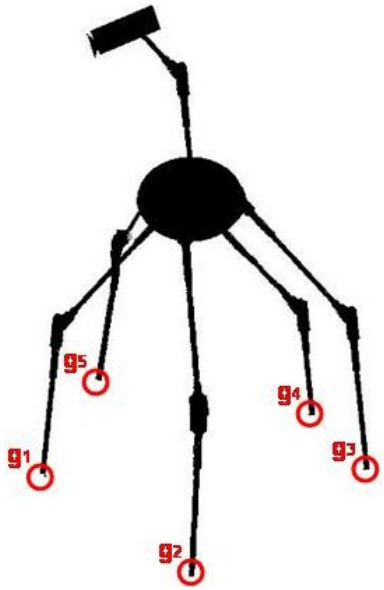


Figure 3.2: General structure of a legged robot, highlighting the points of contact with the supporting surface.

Coordinate reference systems In order to propagate the correction computed to minimize the error detected on the visual features it is first necessary to define the relevant coordinate reference systems that represent the different points of view for the representation of the information coexisting in the robot. We denote I_j the generic reference system j, and ${}_iI_j$ the transformation from generic reference system I_j to generic reference system I_i.

The three coordinate reference systems of interest in our application, ilustrated in figure 3.3, are:

- The fixed reference system whose origin lies on the ground, I_g. Previous works in the literature [31] assume a known world reference system. We

define this system anchored to one of the basic ground support points defined later.

- The body coordinate reference system, I_b, whose origin is the geometrical center of the robot's body. It is assumed that all the readings from the system configuration (i.e. leg configurations) are provided in this frame of reference.

- The camera reference system, I_c.

Having these three basic reference systems we have to define the transformation matrices between them. Notice that every transformation is defined on the parameters of a subset of the robot joints, i.e. transformations between I_g and I_b depend on the articulation joints of the legs with the support points θ_l, while transformations between I_b and I_c depend on the chain of joints from the body center up to the camera θ_u.

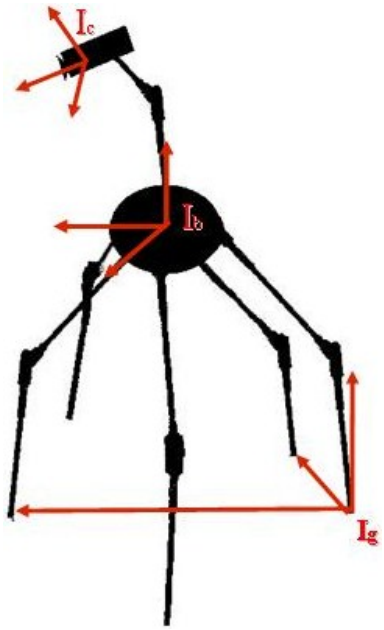


Figure 3.3: Reference systems of the robot.

3.2.1 Leg's degrees of freedom

Each leg has a chain of articulations, as shown in figure 3.4. The leg's degrees of freedom are used to detect the the robot support points, so we introduce this concept first.

Support points

The support points are the points of the robot's legs that determine the supporting plane where it is standing on. From the point of view of the robot's body center, the supporting plane has an apparent motion resulting from the variation in the joint's angles, while, in fact, the physical reality is that the supporting plane remains fixed and the robot changes its pose as a result of the variations in the leg configurations.

Each leg has, at most, a unique support point, and, according to the restriction that the robot must be standing, at least three of the legs must have their supporting points in contact with the ground. In order to determine which ones are the supporting points, we proceed as follows:

1. We obtain the tip position of each leg in the body center coordinate reference system. This position can be computed from the joint angles of each leg articulation.

2. We compute the hypothetical supporting plane defined by each combination of three leg tip points .

3. We discard hypothetical supporting planes for which at least one leg tip is *below* it.

First we compute the position of the leg tip using the coordinate system transformations in the articulation chain from the leg tip up to the body center. These transformations are described in terms of rotation and translation matrices in homogeneous coordinates. For each joint we define a rotational matrix determined by its angles, and for every rod connecting a pair of consecutive joints we define a translation matrix. Figure 3.4 depicts the abstract geometrical representation of this sequence of coordinate reference systems and matrix transformations that link the body center and the leg tip. In homogeneous coordinates the transformation from the reference system whose origin is the tip of a leg into the body center reference system can be described as the following product of elemental transformation matrices:

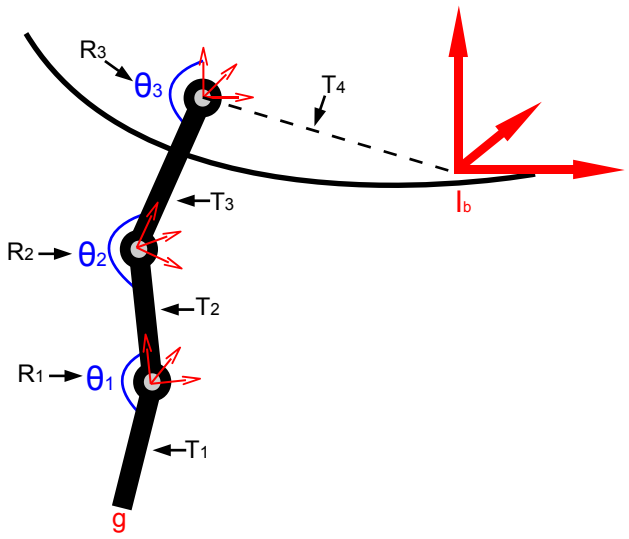


Figure 3.4: Geometry of the leg's articulations.

$$\begin{pmatrix} \mathbf{g} \\ 1 \end{pmatrix} = (\mathbf{T}_{n+1} \cdot \cdot \mathbf{R}_n . \mathbf{T}_n \cdots \mathbf{R_1} . \mathbf{T}_1) . \begin{pmatrix} \mathbf{0} \\ 1 \end{pmatrix}, \tag{3.1}$$

where $\mathbf{R}_k$ is the rotation matrix corresponding to the k-th leg articulation from tip to the body center, being n the number of articulations and $\mathbf{T}_{k-1}$ the translation matrix corresponding to the rod attaching the $(k-1)$-th and the k-th articulations. Translation matrix $\mathbf{T}_1$ corresponds to the translation from the tip to the first articulation, while translation matrix $\mathbf{T}_{n+1}$ corresponds to the translation from the last articulation to the body center reference system. For a given robot's leg, we denote $\mathbf{L}$ the transformation giving the leg's tip:

$$\mathbf{L} = (\mathbf{T}_{n+1} \cdot \mathbf{R}_n . \mathbf{T}_n \cdots \mathbf{R_1} . \mathbf{T}_1).$$

For a given combination of leg tip points we can compute the parameters of the hypothetical support plane equation,

$$\pi : ax + by + cz + d = 0,$$

then we evaluate to which hemisphere belong the remaining points that have not been taken into account to build the hypothetical support plane equation; if for any one of these leg tip points $\mathbf{g}$ we find:

$$\mathbf{g} : a\mathbf{g}_x + b\mathbf{g}_y + c\mathbf{g}_z + d < 0,$$

where $\mathbf{g}_x$ denotes the x coordinate of point $\mathbf{g}$, that means that this point is under the hypothetical support plane and therefore it does not correspond to the ground surface; in contrast, if we find:

$$\mathbf{g} : a\mathbf{g}_x + b\mathbf{g}_y + c\mathbf{g}_z + d > 0,$$

it means that this point is above the hypothetical support plane and therefore it may correspond to the ground surface. Once we find a hypothetical support plane for which all the remaining tip points are above it, we can declare it as the support plane, subject to the following stability condition. In order for the robot to be standing in a stable pose, the projection of the body mass center, according to the direction of gravity, must lie inside of the triangle defined by the basic support points. This condition is illustrated in figure 3.5. The set of basic support points $\mathbf{G}_\pi$ is therefore composed of the three tip points whose corresponding plane is below all the remaining tip points and whose corresponding triangle contains the projection of the body mass center. We define, in the body reference system I_b, the set of basic support points as:

$$\mathbf{G}_\pi = \begin{pmatrix} \mathbf{g}_1 \\ \mathbf{g}_2 \\ \mathbf{g}_3 \end{pmatrix} \in \mathbb{R}^9. \tag{3.2}$$

The extended set of support points is defined as:

$$\mathbf{G}_e = \{\mathbf{g}\ s.t.\ |a\mathbf{g}_x^q + b\mathbf{g}_y + c\mathbf{g}_z + d| < tol\},$$

where tol is a tolerance limit for the distance between the ground plane and a tip point in order to accept it as a support point.

3.2.1.1 Transformation between ground and body systems

In order to define the coordinate transformation ${}_bI_g$ between the ground reference system I_g and the body reference system I_b, we first build, in the reference system I_b, the expression of the director vectors which define the axes of I_g, and then we give the expression of the transformation of the axes of I_g into those of I_b.

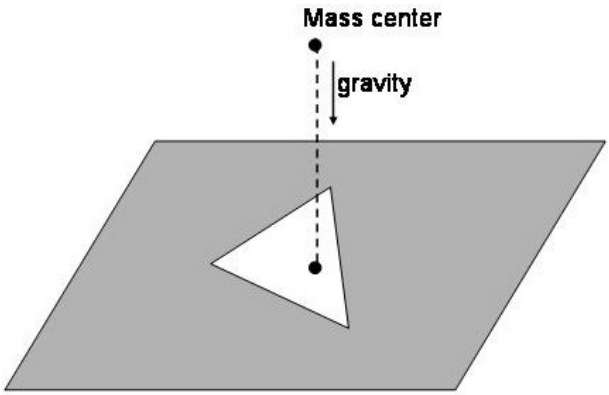


Figure 3.5: Stability Condition to determine the basic support points on the ground plane.

We start considering the three basic support points positions $\mathbf{G}_\pi$. We arbitrarily define the tip point $\mathbf{g}_1$ as the origin of I_g; the vectors $\overrightarrow{\mathbf{g}_1\mathbf{g}_2}$ and $\overrightarrow{\mathbf{g}_1\mathbf{g}_3}$ define the direction of the two first reference axes of I_g and we built the third vector as their cross product. Notice that the axes of I_g lying on the ground plane may not be orthogonal.

The first component of the transformation ${}_bI_g$ is the axes transformation matrix $\mathbf{R}_0$, built from the three director vectors defining the axes of I_g:

$$R_0 = \begin{pmatrix} \mathbf{g}_2 - \mathbf{g}_1 & \mathbf{g}_3 - \mathbf{g}_1 & (\mathbf{g}_3 - \mathbf{g}_1) \times (\mathbf{g}_2 - \mathbf{g}_1) & 0 \\ & & & \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3.3}$$

and the second is the translation from the origin of I_b to the origin of I_g,

$$T_0 = \begin{pmatrix} I_{3\times 3} & \mathbf{g}_1 \\ 0 & 1 \end{pmatrix}. \tag{3.4}$$

So, composing the two transformations we finally obtain the matrix transformation from I_g to I_b,

$${}_bI_g = \mathbf{T}_0\mathbf{R}_0. \tag{3.5}$$

3.2.2 Upper body degrees of freedom

After building the expression of the relationship between the robot's body reference system I_b and the leg's tip, we have to determine the relationship

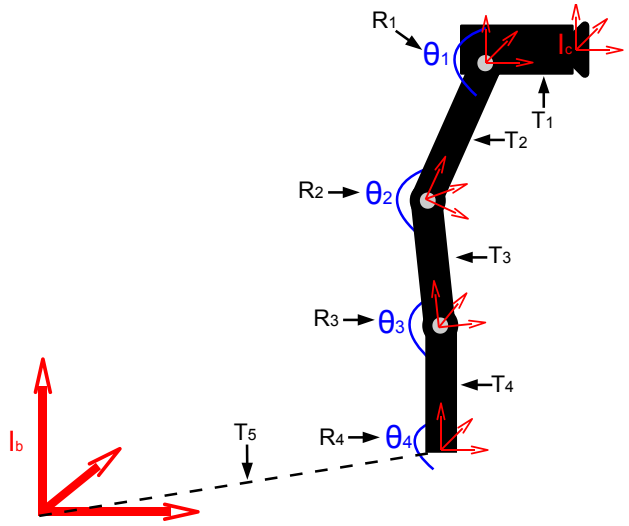


Figure 3.6: Upper body articulations connecting the camera and the body.

between the body reference system and the camera reference system I_c. We construct the transformation applying the rotational and translational matrices of the joints and rods that connect the camera reference system I_c to the robot body reference system I_b.

Transformation between camera reference system and body reference system In homogeneous coordinates, the transformation ${}_bI_c$ from the camera reference system I_c to the body reference system I_b can be done through the composition of the elemental transformations that go from I_b to I_c, as illustrated in figure 3.6. The resulting matrix composition is:

$$ {}_bI_c = (\mathbf{T}_{m+1} \cdot \mathbf{R}_m \cdot \mathbf{T}_m \cdots \mathbf{R_1} \cdot \mathbf{T}_1), $$

where $\mathbf{R}_k$ is the rotation matrix corresponding to the k-th articulation from the origin of the camera system to the origin of the body system, being m the number of articulations and $\mathbf{T}_{k-1}$ the translation matrix corresponding to the rod attaching the $(k-1)$-th and the k-th articulations. Translation matrix $\mathbf{T}_1$ corresponds to the translation from the camera center to the first articulation, while translation matrix $\mathbf{T}_{n+1}$ corresponds to the translation from the last articulation to the center of the body reference system.

We denote $\boldsymbol{\theta}_u$ the angles of the articulation joints in the chain connecting the body reference system to the camera reference system.

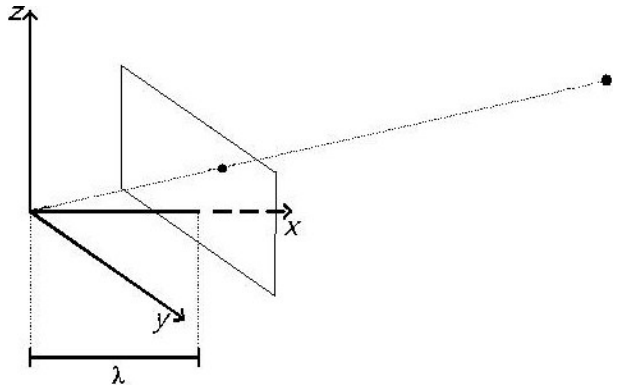


Figure 3.7: Projection of a point on the image plane.

3.2.3 Image features

Visual Servoing's stated goal is to bring the image feature values to the desired target values. We denote the vector of image feature parameters as $\mathbf{s}$ and its desired value as $\mathbf{s}^*$. But these parameters must be expressed in terms of the robot's degrees of freedom, in order to be able to define the Jacobian matrix that characterizes the image feature changes in response to changes in each articulation position.

The camera reference system allows to express the image feature positions according to the robot's camera intrinsic and extrinsic parameters. Figure 3.7 shows the projection of a point on the camera's image plane.

The image plane coordinates of the visual feature $\mathbf{s}_i = (u_i, v_i)^T$ are determined by its position in the camera system ${}^c\mathbf{s}_i = (x_i, y_i, z_i)^T$, according to the following projective equation:

$$s_i = \begin{pmatrix} u_i \\ v_i \end{pmatrix} = \frac{\lambda}{x_i} \begin{pmatrix} y_i \\ z_i \end{pmatrix} = \phi({}^c\mathbf{s}_i). \tag{3.6}$$

The visual features will be, then, expressed in terms of their positions in the camera reference system I_c. We assume that the object is static respect to the ground reference system I_g, therefore we can obtain the image plane coordinates of the visual features from their coordinates ${}^g s$ in I_g using the transformation ${}_cI_g$ between I_g and I_c, defined as the composition of the previously defined transformations ${}_cI_b$ and ${}_bI_g$.

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \phi({}_cI_g({}^g s_i)) \tag{3.7}$$

3.2.4 Construction of the robot's Jacobian matrices

In order to construct the global system's Jacobian matrix $J_{s\theta}$, that relates the variations of the robot degrees of freedom with the variations of the visual feature projections in the image plane, we need to obtain the relations between the components of the robot body and the image plane. We start building the Jacobian matrix that defines the dependence of the image features on the camera reference system J_{sc}. Then we obtain the Jacobian matrices for the dependence of the camera reference system on the upper body articulations $J_{c\theta_u}$ and the basic support points positions $J_{c\pi}$. Then we define the Jacobian matrix that relates the basic support points positions with the positions of all the support points $J_{\pi e}$. Finally, we obtain the Jacobian matrix for the dependence of the extended support positions with the articulations of their legs $J_{e\theta_e}$.

Dependence of image features on the camera J_{sc}

Deriving the image projection in equation 3.6 we get the following variational relation for an image point $\mathbf{s}_i$:

$$\begin{pmatrix} \dot{u}_i \\ \dot{v}_i \end{pmatrix} = \begin{pmatrix} \frac{-\lambda.y_i}{x_i^2} & \frac{\lambda}{x_i} & 0 & 0 \\ \frac{-\lambda.z_i}{x_i^2} & 0 & \frac{\lambda}{x_i} & 0 \end{pmatrix} \cdot \begin{pmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{z}_i \\ 0 \end{pmatrix}. \tag{3.8}$$

We denote J_{s_ic} the Jacobian matrix of each feature point:

$$J_{s_ic} = \begin{pmatrix} \frac{-\lambda.y_i}{x_i^2} & \frac{\lambda}{x_i} & 0 & 0 \\ \frac{-\lambda.z_i}{x_i^2} & 0 & \frac{\lambda}{x_i} & 0 \end{pmatrix}.$$

In order to build the Jacobian matrix J_{sc} that relates the variations of the features in camera space with their image projections, we aggregate the individual feature Jacobian matrices J_{s_ic} into the following block diagonal matrix:

$$J_{sc} = \begin{pmatrix} J_{s_1c} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & J_{s_kc} \end{pmatrix}. \tag{3.9}$$

Therefore, J_{sc} defines a lineal transformation from variations of feature point positions in the camera reference system I_c into variations of the image feature vector $\mathbf{s}$,

$$\Delta \mathbf{s} \simeq J_{sc} \cdot \Delta \left({}^{c}\mathbf{s} \right).$$

Dependence of the basic support points $\mathbf{G}_\pi$ on the extended support points $\mathbf{G}_e$: $J_{\pi e}$

When the extended support points change, the ground plane may change and so the basic ground points change. The following matrix equation relates variations in the vector of the basic support points $\triangle \mathbf{G}_\pi$with variations in $\triangle \mathbf{G}_e$:

$$\triangle \mathbf{G}_\pi = J_{\pi e} \triangle \mathbf{G}_e,$$

which can be expanded as follows:

$$\begin{pmatrix} \triangle \mathbf{g}_1^\pi \\ \triangle \mathbf{g}_2^\pi \\ \triangle \mathbf{g}_3^\pi \end{pmatrix} = \begin{pmatrix} M_{11} & \cdots & M_{1n} \\ M_{21} & \cdots & M_{2n} \\ M_{31} & \cdots & M_{3n} \end{pmatrix} \begin{pmatrix} \triangle \mathbf{g}_1^e \\ \vdots \\ \triangle \mathbf{g}_n^e \end{pmatrix}, \tag{3.10}$$

where $\triangle \mathbf{G}_\pi = (\triangle \mathbf{g}_1^\pi, \triangle \mathbf{g}_2^\pi, \triangle \mathbf{g}_3^\pi)^T$, $\triangle \mathbf{G} = (\triangle \mathbf{g}_1^e, \ldots, \triangle \mathbf{g}_n^e)^T$ and $J_{\pi e} \in \mathbb{R}^{12 \times 4n}$ is the Jacobian matrix of equation 3.10, and the building block matrices of size 4×4, are defined as follows:

- $M_{ij} = \mathbf{I}_{4\times 4}$, if the tip of the j-th leg corresponds to the i-th basic support point.
- $M_{ij} = \mathbf{0}$, if th tipo of the j-th leg does not correspond to the i-th basic support point.

Dependence of camera coordinates on upper articulations and extended support points J_{cp}

For the derivation of the expresion of J_{cp} we will follow a top-down approach, according to figure 3.1.

The feature positions in the camera reference system could be expressed as a function of their coordinates in the ground reference system. This transformation depends on the upper body articulations and the basic support points,

according to equation 3.7. By deriving this equation, we get the Jacobian matrix that relates the variations in the feature positions in the camera reference system I_c with the variations in the upper body articulations and the basic support points positions:

$$J_{cp} = \frac{\delta\left({}^{c}\mathbf{s}\right)}{\delta \mathbf{p}_{\pi}} = \frac{\delta({}_{c}I_{b} \cdot_{b} I_{g}\left({}^{g}\mathbf{s}\right))}{\delta \mathbf{p}_{\pi}}, \tag{3.11}$$

where $\mathbf{p}_{\pi}$ is the following vector of upper joints and basic support points positions:

$$\mathbf{p}_{\pi} = \begin{pmatrix} \boldsymbol{\theta}_{u} \\ \mathbf{G}_{\pi} \end{pmatrix}. \tag{3.12}$$

Using the chain rule, we rewrite equation 3.11:

$$J_{cp} = \frac{\delta({}_{c}I_{b})}{\delta \mathbf{p}_{\pi}}({}_{b}I_{g})\left({}^{g}\mathbf{s}\right) + ({}_{c}I_{b})\frac{\delta({}_{b}I_{g})}{\delta \mathbf{p}_{\pi}}\left({}^{g}\mathbf{s}\right). \tag{3.13}$$

As ${}_{c}I_{b}$ is a function only of $\boldsymbol{\theta}_{u}$ (the camera articulations) and ${}_{b}I_{g}$ is a function only of $\mathbf{G}_{\pi}$, we define the following independent Jacobians from equation 3.13:

$$J_{c\theta_{u}} = \frac{\delta({}_{c}I_{b})}{\delta \boldsymbol{\theta}_{u}}({}_{b}I_{g})({}^{g}\mathbf{s}), \tag{3.14}$$

$$J_{c\pi} = ({}_{c}I_{b})\frac{\delta({}_{b}I_{g})}{\delta \mathbf{G}_{\pi}}({}^{g}\mathbf{s}), \tag{3.15}$$

where $J_{c\theta_{u}}$ defines the dependences of the features in the camera reference system on the upper body articulations and $J_{c\pi}$ defines the dependences of the features in the camera reference system on the basic support points.

If we make use of matrix $J_{e\pi}$, defined in equation 3.10, we can define the dependences of the camera on the extended support points:

$$J_{ce} = J_{c\pi} \cdot J_{\pi e}. \tag{3.16}$$

The Jacobian matrix J_{cp} can be constructed as the aggregation of the Jacobian matrices $J_{c\pi}$ and $J_{\pi e}$, corresponding to the kinematics in two orthogonal subspaces, into the following block diagonal matrix:

$$J_{cp} = \begin{pmatrix} J_{c\theta_{u}} & 0 \\ 0 & J_{ce} \end{pmatrix}. \tag{3.17}$$

The dependence of the variations in the camera feature positions on the variations in the upper body degrees of freedom and the extended support points positions can be summarized by:

$$\Delta\left({}^{c}\mathbf{s}\right) \simeq J_{cp}\mathbf{\Delta p}, \tag{3.18}$$

being $\mathbf{p}$ the following vector of upper joints and extended support point positions:

$$\mathbf{p} = \begin{pmatrix} \theta_u \\ \mathbf{G}_e \end{pmatrix} \tag{3.19}$$

Dependence of p on the robot's articulation angles: $J_{p\theta}$

We can decompose the Jacobian matrix $J_{p\theta}$, which defines the dependence of $\mathbf{p}$ on the robot's degrees of freedom, into two Jacobian matrices, one which relates the extended support points and the leg's articulations $J_{e\theta}$ and the other which is the identity matrix for the upper body articulations. To build $J_{e\theta}$ we start modelling the changes in the extended support point coordinates in response to the changes in the degrees of freedom of the corresponding leg, as follows:

$$\Delta\mathbf{g}_i^e \simeq J_i \cdot \Delta\boldsymbol{\theta}_i, \tag{3.20}$$

where J_i is the Jacobian matrix of equation 3.1, relating the variations of the i-th extended support point as function of variations of the leg joint angles $\boldsymbol{\theta}_i = (\theta_{i1}, \theta_{i2}, \ldots, \theta_{im_i})$ in the leg corresponding to this i-th extended support point. The size of each J_i matrix is $4 \times m_i$, being m_i the number of joints of leg i.

Aggregating the extended support point Jacobians into a diagonal block matrix, we get the following equation:

$$\begin{pmatrix} \triangle\mathbf{g}_1^e \\ \vdots \\ \triangle\mathbf{g}_n^e \end{pmatrix} = \begin{pmatrix} J_1 & 0 & 0 & 0 \\ 0 & J_2 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & J_n \end{pmatrix} \begin{pmatrix} \triangle\boldsymbol{\theta}_1 \\ \triangle\boldsymbol{\theta}_2 \\ \vdots \\ \triangle\boldsymbol{\theta}_n \end{pmatrix}, \tag{3.21}$$

which can be expressed in matrix form as follows:

$$\triangle\mathbf{G}_e = J_{e\theta}\Delta\boldsymbol{\theta_e}, \tag{3.22}$$

where $\boldsymbol{\theta_e}$ is the vector composed of all the joint angles of the legs corresponding to all the extended support points.

In order to obtain a single matrix that relates the variations of $\mathbf{p}$ in all the articulation joints of the robot, we define the following diagonal block matrix:

$$J_{p\theta} = \begin{pmatrix} \mathbf{I}_{m\times m} & 0 \\ 0 & J_{e\theta} \end{pmatrix}, \tag{3.23}$$

where m is the number of upper body articulations.

The dependence of variations in vector $\mathbf{p}$ on variations in the robot degrees of freedom, can be summarized as:

$$\boldsymbol{\Delta}\mathbf{p} \simeq J_{p\theta}\boldsymbol{\Delta\theta}. \tag{3.24}$$

Image Jacobian matrix

Finally, we define the full Jacobian matrix that models the dependence of the image features on all the degrees of freedom of the robot by composing the Jacobian matrices 3.9, 3.17 and 3.24 obtained previously:

$$\Delta\mathbf{s} = (J_{sp}J_{p\theta})\Delta\theta, \tag{3.25}$$

where $J_{sp} = J_{sc}J_{cp}$. The global Jacobian matrix is defined as:

$$J_{s\theta} = J_{sp}J_{p\theta}. \tag{3.26}$$

The equation 3.25 is thus rewritten in the following way:

$$\Delta s = J_{s\theta}\Delta\theta. \tag{3.27}$$

3.3 Inverse kinematics

The stated goal of Visual Servoing is to determine the instantaneous changes of each of the robot's degrees of freedom that are needed in order to bring the image feature parameters to the desired values (positions).

In order to determine the velocity at each robot's degrees of freedom that will give the desired result, we should obtain the inverse of the $J_{s\theta}$ matrix in equation 3.27. In general, this matrix is not invertible, because it is underconstrained.

The general solution by minimum least squares is to use the seudoinverse of $J_{s\theta}^{+}$ in the following way:

$$\dot{\boldsymbol{\theta}} = J_{s\theta}^{+}\dot{\mathbf{s}} + (I - J_{s\theta}^{+}J_{s\theta})\mathbf{w}, \tag{3.28}$$

where $\mathbf{w}$ is an arbitrary vector of R^{m+3n}, being m the number of upper body joints and n the number of extended support points.

In general, $(I - J_{s\theta}^{+}J_{s\theta})\mathbf{w} \neq 0$ and all the vectors of the form $(I - J_{s\theta}^{+}J_{s\theta})\mathbf{w}$ belong to the kernel of the transformation associated to $J_{s\theta}$.

This solution minimizes the norm of the visual error:

$$\left\| \dot{s} - (J_{s\theta})\dot{\theta} \right\|. \tag{3.29}$$

But this solution does not take into account the restriction of keeping the distances between supporting points constant, which is the necessary condition to mantain the ground reference system invariant.

So, we need to determine how the variations in the robot's degrees of freedom affect the distances between the supporting points.

We define $\mathbf{d}$ as the vector containing the distances between support points:

$$\mathbf{d} = \begin{pmatrix} d_1 \\ \vdots \\ d_{n(n-1)} \end{pmatrix}. \tag{3.30}$$

We get the Jacobian matrix that relates the changes in this vector to the changes in the extended support points.

$$J_{de} = \begin{pmatrix} \frac{\delta d_1}{\delta \mathbf{e}_1} & \cdots & \frac{\delta d_1}{\delta \mathbf{e}_n} \\ \vdots & \vdots & \vdots \\ \frac{\delta d_n}{\delta \mathbf{e}_1} & \cdots & \frac{\delta d_n}{\delta \mathbf{e}_n} \end{pmatrix}. \tag{3.31}$$

In order to model the invariance against the variations in the upper body's degrees of freedom, we extend the Jacobian matrix J_{dp} from J_{de}:

$$J_{dp} = \begin{pmatrix} 0_{n\times m} & 0 \\ 0 & J_{de} \end{pmatrix}. \tag{3.32}$$

Finally, the dependences of the distances among the extended support points are resumed in the following equation:

$$\Delta\mathbf{d} \simeq J_{dp} \cdot \begin{pmatrix} \Delta\theta_u \\ \Delta\mathbf{G_e} \end{pmatrix}. \tag{3.33}$$

Using J_{dp}, J_{pc} and J_{sc} we can predict the variations in the distances between support points as a function of the image features variations. This relation is formalized as follows:

$$\Delta\mathbf{d} = [J_{dp}J_{pc}J_{sc}^{+}]\Delta\mathbf{s}.$$

If we want to define a control system that preserves $\mathbf{d}$ we must project the obtained movements into the null space of $[J_{dp}J_{pc}J_{sc}^{+}]$, therefore we define the following control rule for the movements in the extended support points:

$$\Delta\mathbf{p^1} = [(I - J_{dp}^{+}J_{dp})(J_{pc}J_{sc}^{+})]k_s\Delta\mathbf{s}. \tag{3.34}$$

However, due to the limitations of the linear aproximations, the real movements in the support points produce undesired variations in $\mathbf{d}$. Moreover, the errors of the robot positioning system produce additional variations in $\mathbf{d}$. Therefore, some corrective actions for repositioning the support points are required:

$$\Delta\mathbf{p^2} = [(I - J_{sp}^{+}J_{sp})J_{dp}^{+}]k_d\Delta\mathbf{d}, \tag{3.35}$$

where we project the movements obtained by applying the seudoinverse of Jacobian J_{dp} to the error in the distances between extended support points positions into the null space of J_{sp}^{+}.

Finally, we combine equations 3.34 and 3.35 in order to obtain a control law that moves the articulations maintaining the ground reference system invariant while moving the image features to the desired ones. This global control law is defined as follows:

$$\Delta\theta = J_{p\theta}^{+}\{(I - J_{dp}^{+}J_{dp})(J_{ps})]k_s\Delta\mathbf{s} + (I - J_{sp}^{+}J_{sp})J_{dp}^{+}k_d\Delta\mathbf{d}\}, \tag{3.36}$$

being k_s and k_d the speed constants for image control (equation 3.34) and extended support points positions control (equation 3.35), respectively.

This equation allows us to determine the variations on the robot's degrees of freedom to get the desired configuration of the image. However, this equation is unrestricted and may drive the robot into unstable configurations, that is, to articulation configurations out of the region of stable poses in configuration space. Stable poses are characterized by the condition illustrated in figure 3.5. When this condition does not hold or the projection point is too close to the polygon boundary we restrict the Visual Servoing to

the upper body degrees of freedom, using the transformation ${}_gI_c$ instead of ${}_bI_c$ to construct a reduced Jacobian $J_{s\theta_u}$ that relates the image features to the upper body degrees of freedom.

3.4 Experimentation with an Aibo ERS-7 robot

In this section we apply the ideas of Visual Servoing developed in previous sections to build a formal model to perform visual tracking of a ball using all the degrees of freedom of a Sony's Aibo ERS-7 robot. We have also performed real life experiments under controlled conditions to asses the applicability of our approach, reporting the quantitative results of such experiments. Nowadays visual tracking solutions for this kind of robots inspired in the Visual Servoing approach only move the head effectors or perform motions programmed on the basis of high level primitives (walk, turn) involving complex leg motions. In this work we take into account all the effectors which can affect the resulting image. We construct, from the description of the robot, the matrix that describes the direct kinematics of the robot and obtain the low level control commands by applying the inverse kinematics.

Figure 3.8 illustrates the main feedback loop in image-based Visual Servoing with the Aibo.

In the RoboCup robot soccer matches some Visual Servoing approaches [59, 61] have been implemented in the Aibo robot to track the ball. However, these approaches are limited to the movement of the head effectors in order to keep the ball inside the robot camera field of view. The space in which the ball can be followed is restricted by the robot's body pose.

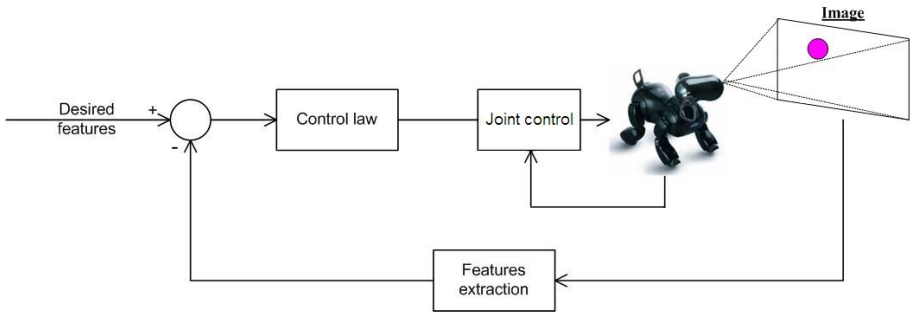


Figure 3.8: Visual Servoing feedback loop

In this section we address the precise task of maintaining the playing ball in the center of the robot's camera image. The only visual feature considered is the center of the ball region in the image identified by the color detection routines which are primitives in the robot's basic control software. For program development, we have profited from the Carnegie Mellon University's SDK [84] and the SONY's SDK [10]. The image error is the distance in the image space between the image center and the centroid of the blob corresponding to the ball.

In order to follow the construction of the image Jacobian matrix defined in section 3.2.4, we need to define the following Jacobian matrices:

- J_{sc}: dependence of the image features on the camera reference system.
- $J_{c\theta_u}$: dependence of the camera reference system on the upper body degrees of freedom.
- $J_{c\pi}$: dependence of the camera reference system on the basic support points.
- $J_{e\theta_e}$: dependence of the extended support points on their respective leg's articulations.

We will start defining the image feature vector from the ball parameters, then we will define the Jacobian matrices building the direct linear model for the kinematics of the robot and we will apply the inverse kinematics. Finally, we end this section with some discussion of the physical experimentation, the observed robot behavior and future work lines.

3.4.1 Image feature vector

The stated task goal is to bring the ball to the image center, so the target value of the feature vector are the image center coordinates and the observed features from the real world are the image coordinates of the ball region center and the observed ball diameter.

The camera reference system is fixed on the robot head and it is the frame of reference to define the ball position for the visual sub-system of the robot. In figure 3.9 the projection of the ball on the robot's camera plane is presented.

The image feature parameter vector, $\mathbf{s}$, is determined by the ball position in the camera system, according to the following relation:

$$\mathbf{s} = \begin{pmatrix} u \\ v \end{pmatrix} = \phi({}^c\mathbf{ball}) = \frac{\lambda}{{}^c\mathbf{ball}_x} \begin{pmatrix} {}^c\mathbf{ball}_y \\ {}^c\mathbf{ball}_z \end{pmatrix}. \tag{3.37}$$

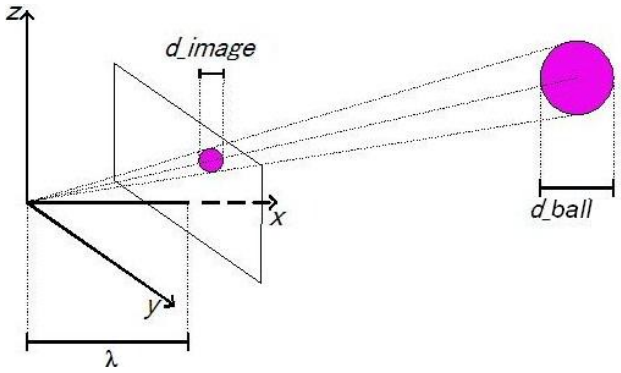


Figure 3.9: Ball projection on the camera plane

The value of $\mathbf{s}$ is estimated from the segmented image as the average position of the pixels detected as corresponding to ball pixels. This process uses the Aibo's basic image segmentation software, which sometimes produces (many) false-positives. These false-positives introduce aditional errors in the Visual Servoing.

The ball position in the camera reference system is determined by estimating the distance from the camera to the ball. In order to get the distance we have to take into account the real diameter of the ball and its diameter in the image plane. The following equation shows this relation:

$$distance = \lambda \frac{1}{2} \frac{diam_r}{\tan(\frac{1}{2} diam_i)}, \tag{3.38}$$

where $diam_r$ is the real diameter of the ball and $diam_i$ is the measured diameter of the ball projection in the image plane. Using the estimated distance we can compute the ball center position in the camera reference system:

$${}^c ball = distance \begin{pmatrix} cos(v).cos(u) \\ sin(u) \\ sin(v) \end{pmatrix}. \tag{3.39}$$

The features are expressed in terms of the ball position in the system I_c. Assuming that the ball was fixed respect to I_g, we could obtain the feature vector expressed in function of the head robot articulations and the support point positions, using the ball position in I_g and the transformation between I_g and I_c.

$$\begin{pmatrix} u \\ v \end{pmatrix} = \phi({}_cI_g({}^gball)) \tag{3.40}$$

3.4.2 Direct kinematics

We build the Aibo kinematics model as a transformation from the ground supporting plane to the head coordinate system, composing the diverse transformations that correspond to the degrees of freedom of the legs and the head.

3.4.2.1 Degrees of freedom of the legs

As illustrated in figure 3.10, the robot's feet and knees are the possible robot support points, therefore we need to be able to determine their 3D coordinates at any time.

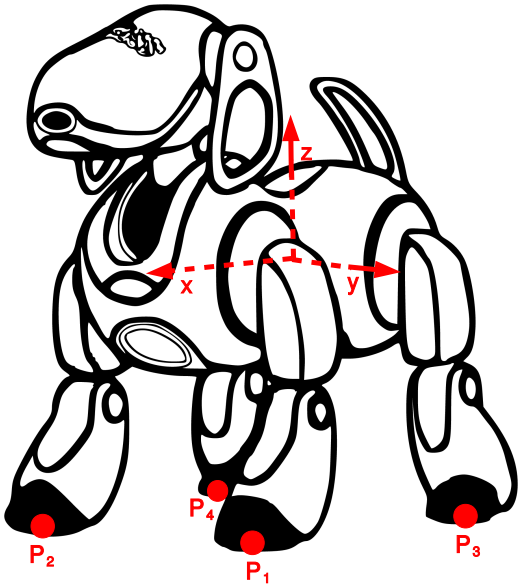
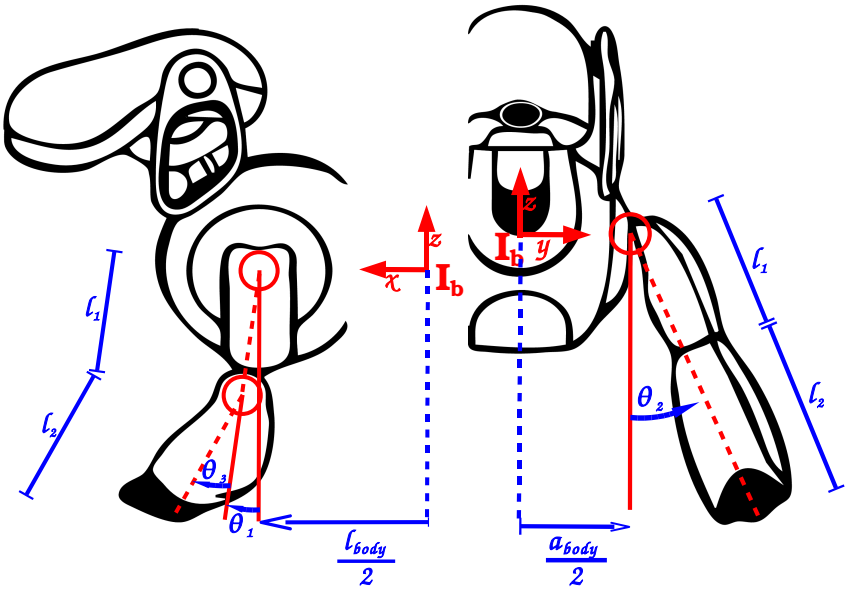


Figure 3.10: Points of contact with the supporting surface



(a) Lateral view. (b) Frontal view.

Figure 3.11: Geometry of the leg's articulations.

Each leg has three articulations, as shown in figure 3.11. The Aibo possesses an inertial sensor than gives us information of the gravity direction in the body reference system which can be used to evaluate the stability condition of the hypotetical support plane.

Each leg has a unique support point that can be the foot or the knee. According to the restriction that the robot must be standing, at least three of the legs must have their support points in contact with the ground; therefore all the feasible combinations of feet and knees give us 32 hypotetical support planes.

It is necessary to determine the positions of the feet and knees relative to the body center in function of the articulation angles. In figure 3.11 we show the geometry of the legs.

For the front left leg we find the foot position using the following sequence of coordinate system transformations:

- T_1: Translation along the z-axis of length l_1.
- R_1: Clockwise rotation about y-axis by angle θ_1.

- R_2: Counterclockwise rotation about x-axis by angle θ_2.
- R_1: Clockwise rotation about y-axis by angle θ_3.
- T_2: Translation along the z-axis of length l_2.
- T_b: Translation along the x-axis of length $\frac{1}{2}l_{body}$, being l_{body} the robot body length.
- T_a: Translation along the y-axis of length $\frac{1}{2}a_{body}$, being a_{body} the robot body width.

In homogeneous coordinates the following transformation of the body center gives us the positions of the foot:

$$\begin{pmatrix} X_p \\ Y_p \\ Z_p \\ 1 \end{pmatrix} = (T_a \cdot T_b \cdot R_1 \cdot R_2 \cdot T_1 \cdot R_3 \cdot T_2). \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

This equation for the robot's front left leg can be easily adapted to get the possitions of the other three leg's feet. To compute the coordinates of each knee in the body reference system we only have to remove $R_3 \cdot T_2$ from the above sequence of tranformations.

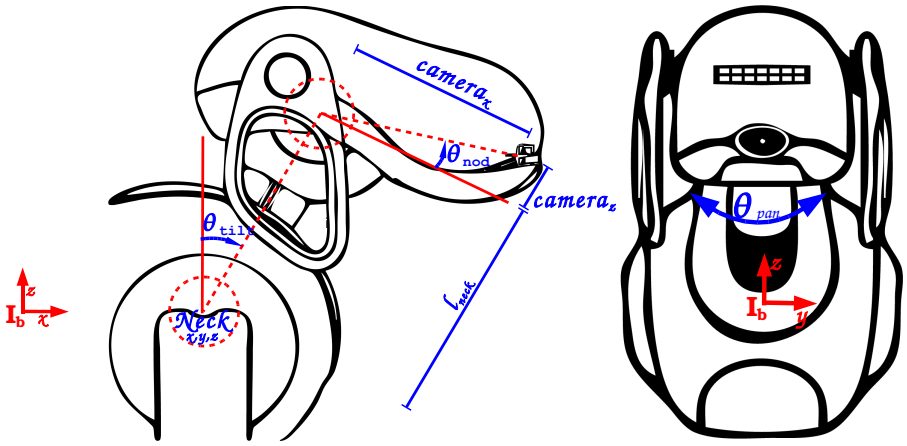
3.4.2.2 Head's degrees of freedom

The Aibo ERS-7 has three degrees of freedom in the head. That introduces ambiguity in the control trajectories needed to track the ball trajectory.

Figure 3.12 shows the two tilt degrees of freedom of the Aibo, denoted θ_{tilt} and θ_{nod}. The first head tilt degree of freedom corresponds to the neck base pivoting along part of the dog chest, while the second one allows the head to move vertically using as the rotation center the joint between the neck and the head. The third degree of freedom, called θ_{pan}, allows a perpendicular rotation to the previous ones, moving the head from side to side.

3.4.2.3 Coordinate reference systems

We define the relevant reference systems as explained in section 3.2. In order to obtain the ball position expressed in the I_g ground system it is necessary



(a) Lateral view. (b) Frontal view.

Figure 3.12: Head's degrees of freedom.

to obtain the transformation matrices between the different systems. These reference systems are illustrated in figure 3.13.

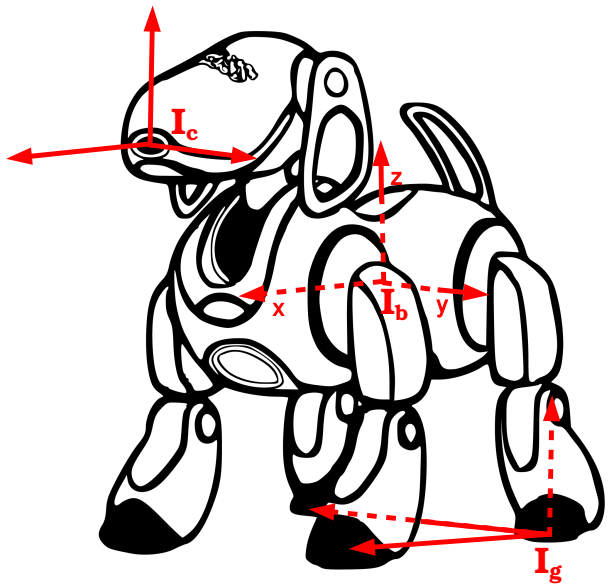


Figure 3.13: Aibo reference systems: I_g, I_b, I_c.

Transformation betweenI_g and I_b In order to define the coordinates change from the ground system I_g to the body system I_b, we define the transformation matrix ${}_bI_g$, as explained in section 3.2.1.1.

The entire transformation uses the basic support point positions: $\mathbf{G}_\pi = (\mathbf{g}_1, \mathbf{g}_2, \mathbf{g}_3)^T$. As we assume that the four legs of the Aibo are on the ground surface, we define the left back leg as $\mathbf{g}_1$, the left front as $\mathbf{g}_2$ and the right back as $\mathbf{g}_3$.

The rotational matrix R and the translational matrix T are obtained as explained in section 3.2.1.1 and then we finally obtain the matrix change from I_g to I_b, composing the two transformations:

$$ {}_bI_g = T \cdot R. \tag{3.41}$$

Transformation betweenI_b and I_c The transformation between these systems is the composition of more elemental transformations. The first transformation is a translation from the camera base to the top of the neck:

$$ T_1 = \begin{pmatrix} 1 & 0 & 0 & camera_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & camera_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3.42}$$

where $camera_x$ and $camera_z$ define the translation from the camera origin to the neck's top joint. Next, we have to take into account the *nod* and *pan* rotations. We call this rotational matrix:

$$ R_1 = \begin{pmatrix} cos(\theta_{pan})cos(\theta_{nod}) & -sen(\theta_{pan}) & -cos(\theta_{pan})sen(\theta_{nod}) & 0 \\ sen(\theta_{pan})cos(\theta_{nod}) & cos(\theta_{pan}) & -sen(\theta_{pan})sen(\theta_{nod}) & 0 \\ sen(\theta_{nod}) & 0 & cos(\theta_{nod}) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.43}$$

Then, we take into account the translation T_2 from the neck top joint to the neck base joint:

$$ T_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & l_{neck} \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3.44}$$

where l_{neck} is the neck length.

Then, we take into account the tilt articulation defining the rotational matrix R_2:

$$R_2 = \begin{pmatrix} cos(\theta_{tilt}) & 0 & -sen(\theta_{tilt}) & 0 \\ 0 & 1 & 0 & 0 \\ sen(\theta_{tilt}) & 0 & cos(\theta_{tilt}) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.45}$$

Finally, we take into account the translation to the body center T_3:

$$T_3 = \begin{pmatrix} 1 & 0 & 0 & neck_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & neck_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{3.46}$$

where $neck_x$ and $neck_z$ are the coordinates of the neck base joint in the body reference system.

The resulting matrix composition ${}_bI_c$ is the transformation from system I_b to system I_c:

$${}_bI_c = T_3 R_2 T_2 R_1 T_1. \tag{3.47}$$

3.4.2.4 Feature Jacobian matrix

Now we will construct the Jacobian matrix that relates the variations of the diverse degrees of freedom of the Aibo with the variations in the image plane.

Dependence of image features on the target object The features must be expressed in terms of the robot's degrees of freedom in order to use the Jacobian to determine the feature sensitivity respect to each articulation position changes.

Deriving the equation 3.37 we get the following relation:

$$\begin{pmatrix} \Delta u \\ \Delta v \end{pmatrix} = \begin{pmatrix} \frac{-\lambda . y_p}{x_p^2} & \frac{\lambda}{x_p} & 0 & 0 \\ \frac{-\lambda . z_p}{x_p^2} & 0 & \frac{\lambda}{x_p} & 0 \end{pmatrix} \begin{pmatrix} \Delta^c ball_x \\ \Delta^c ball_y \\ \Delta^c ball_z \\ 0 \end{pmatrix}. \tag{3.48}$$

According to equation 3.9, we call J_{sc} the Jacobian matrix of equation 3.48. Therefore, J_{sc} defines a lineal transformation from variations in the positions

of the ball represented in I_c into variations of the image features in the image plane.

$$\Delta s \simeq J_{sc} \cdot \Delta({}^{c}ball). \tag{3.49}$$

Dependence of support points on the basic support points The following matrix relates the variations in the extended support points, $\Delta\mathbf{G_e}$, with the variations in the basic support points.

$$\begin{pmatrix} \Delta\mathbf{g}_1 \\ \Delta\mathbf{g}_2 \\ \Delta\mathbf{g}_3 \end{pmatrix} = \begin{pmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \end{pmatrix} \begin{pmatrix} \Delta\mathbf{g}_1^e \\ \Delta\mathbf{g}_2^e \\ \Delta\mathbf{g}_3^e \\ \Delta\mathbf{g}_4^e \end{pmatrix} \tag{3.50}$$

We call $J_{e\pi} \in \mathbb{R}^{12\times 15}$ the Jacobian matrix of equation 3.50, being the M_{ij} matrices defined in section 3.2.4.

The dependence between the variations in the legs articulations with the supporting points positions and the head articulations variations can be summarized in the following equation:

$$\Delta\pi \simeq J_{\pi e} \cdot \Delta e. \tag{3.51}$$

Dependence on support points' articulations The next step is to obtain a linear transformation of the extended support points on the legs' degrees of freedom.

First we observe that, according to which part of the leg is in contact with the ground, there are two possible Jacobian matrices: one for the foot (J_i^f) an another for the knee (J_i^k). We model the changes in the support points coordinates by one of the following equations depending on the support point being a foot or a knee:

$$\Delta f_i \simeq J_i^f \cdot \Delta\boldsymbol{\theta}_i, \tag{3.52}$$

$$\Delta k_i \simeq J_i^k \cdot \Delta\boldsymbol{\theta}_i, \tag{3.53}$$

where $\boldsymbol{\theta}_i$ are the degrees of freedom of leg i.

Aggregating the support points Jacobians into a diagonal block matrix, we obtain the following Jacobian matrix:

$$\begin{pmatrix} \Delta\mathbf{g}_1^e \\ \Delta\mathbf{g}_2^e \\ \Delta\mathbf{g}_3^e \\ \Delta\mathbf{g}_4^e \end{pmatrix} = \begin{pmatrix} M_1 & 0 & 0 & 0 \\ 0 & M_2 & 0 & 0 \\ 0 & 0 & M_3 & 0 \\ 0 & 0 & 0 & M_4 \end{pmatrix} \cdot \begin{pmatrix} \Delta\boldsymbol{\theta}_1 \\ \Delta\boldsymbol{\theta}_2 \\ \Delta\boldsymbol{\theta}_3 \\ \Delta\boldsymbol{\theta}_4 \end{pmatrix} \tag{3.54}$$

This Jacobian matrix receives the name $J_{e\theta_e}$, where M_i is:

- J_i^f, if the support point for the leg i is the foot.
- J_i^k, if the support point for the leg i is the knee.
- $Zero$ (the matrix with all the elements equal 0) if this leg does not have a support point on the ground plane.

The dependence of the extended support points on the legs' degrees of freedom is summarized as follows:

$$\Delta\mathbf{G}_e \simeq J_{e\theta_e}\Delta\boldsymbol{\theta}_e. \tag{3.55}$$

Dependence of image features on basic support points and upper body articulations According to the development done in section 3.2.4, we obtain the matrices that define the relation between the ball position in the camera system and the support points and the head robot articulations by deriving equation 3.40. The Jacobian matrix that relates the variations in the image ball position with the variations in the support points positions is defined as:

$$J_{c\pi} = \frac{{}_cI_{bb}I_g({}^gball)}{\delta\mathbf{G}_\pi}. \tag{3.56}$$

The Jacobian matrix $J_{c\pi}$ is composed by the following derivatives:

$$J_{c\pi} = ({}_cI_b) \begin{pmatrix} \frac{\partial_b I_g({}^gball)}{\partial\mathbf{g}_1} \\ \frac{\partial_b I_g({}^gball)}{\partial\mathbf{g}_2} \\ \frac{{}_b I_g({}^gball)}{\partial\mathbf{g}_3} \end{pmatrix}^T. \tag{3.57}$$

The Jacobian matrix that relates the variations of the ball position in the camera reference system with the variations in the head degrees of freedom is defined as:

$$J_{c\theta_u} = \frac{\delta({}_cI_b)({}_bI_g)({}^g ball)}{\delta\theta_{head}}. \tag{3.58}$$

Only the transformation matrix ${}_cI_b$ depends on θ_{head} and is defined as the composition of elemental matrices, so equation 3.58 can be rewritten as:

$$\frac{\delta({}_cI_b)}{\delta\theta_{head}} = \frac{\delta(T_3R_2T_2R_1T_1)}{\delta\theta_{head}}. \tag{3.59}$$

Matrices T_1, T_2 and T_3 do not depend on the head articulations, therefore only the rotational matrices R_1 and R_2 are derived in order to obtain the derivative of the transformation matrix. Because R_1 depends on nod and pan articulations and R_2 depends on the tilt articulation, we can express the derivative of the transformation matrix from ground system to base system as follow:

$$\frac{\delta({}_cI_b)}{\delta\theta_{head}} = \frac{\delta({}_cI_b)}{\delta\begin{pmatrix}\theta_{pan}\\ \theta_{nod}\\ \theta_{tilt}\end{pmatrix}} = \begin{pmatrix} T_3R_2T_2\frac{\delta R1}{\delta\theta_{pan}}T_1 \\ T_3R_2T_2\frac{\delta R1}{\delta\theta_{nid}}T_1 \\ T_3\frac{\delta R2}{\delta\theta_{tilt}}T_2R_1T_1 \end{pmatrix}^T. \tag{3.60}$$

3.4.3 Inverse kinematics

In order to determine the velocity at each robot's degree of freedom we should apply the pseudoinverse approach of equation 3.36. As we have more degrees of freedom than image features, the problem is over-constrained, because there are not sufficient features to determine the movements in a simple way.

This equation allows us to determine the variations on the robot degrees of freedom to get the desired configuration of the image. However, it is unrestricted and may drive the robot into unstable configurations, that is, to articulation configurations out of the region of stable poses in configuration space. Stable poses are characterized by the existence of a quadruplet of ground support points which fulfill the condition illustrated in figure 3.14. When this does not happen, or the projection point is too close to the quadrange boundary, we restrict the Visual Servoing to the head's degrees of freedom, using the transformation ${}_gI_c$ instead of ${}_bI_c$, to construct a reduced Jacobian J_h that relates the image features to them. Its seudoinverse gives

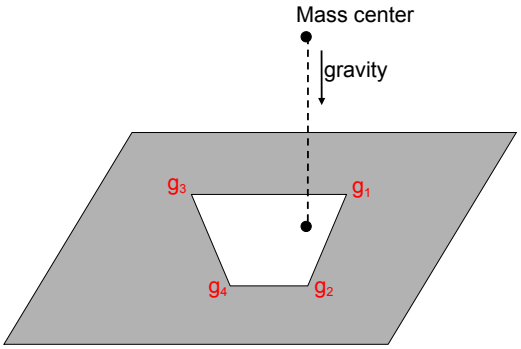


Figure 3.14: Quadruplet of ground support points.

the control for the head's degrees of freedom. This reduced approach has already been applied in [59, 61].

3.4.4 Empirical results on the Aibo

In this section we show the results obtained from applying the approach proposed in this chapter to the visual tracking of a ball using all the degrees of freedom of a Sony's Aibo ERS-7 robot. We examine the behavior of the robot under different controlled experimental settings.

The experiments are based on a definition of a nominal initial position of the Aibo's degrees of freedom, where the Aibo itself gives the origin and orientation for the definition of the world reference system. Then we have defined two experiments aimed to asses the robustness of the proposed Visual Servoing approach under controlled conditions:

- Experiment 1: The ball is placed in a fixed position and the robot performs Visual Servoing to place the ball center in the image plane center, stopping after reaching the goal under some tolerance conditions. The ground plane before the robot is discretized in a specific way to allow for the systematic sampling of the behavior of the robot under varying positions of the ball.

- Experiment 2: The ball is placed in a sequence of positions. The robot is allowed to perform the visual servoing at each position and then the ball is moved to the next position in the sequence. The aim of this experiment is to test the accumulation of errors in the joint controls and

Figure 3.15: Initial configuration of the joints of the Aibo.

the visual tracking subsystem, and the ability of the robot to recover from "uncomfortable" positions.

The initial pose of the Aibo for both experiments is defined as a standing stable position of its body. The values of the joints are shown on table 3.2 and figure 3.15.

Body element	joint	value	joint	value	joint	value
Left front leg	1	0.1	2	0.1	3	0.1
Right front leg	1	0.1	2	0.1	3	0.1
Left back leg	1	0.1	2	0.1	3	0.1
Right back leg	1	0.1	2	0.1	3	0.1
Head	tilt	0.1	pan	0.1	nod	0.1

Table 3.2: Configuration of the joints of the Aibo in the nominal initial pose. The joint values are given in radians.

3.4.4.1 Visual tracking of a static ball

For the experiment, we want to test the response of the robot over an homogeneous distribution of the ball positions within the vision range of the

Aibo. We define $30cm$ as the reference distance between sampling points in the (x, y) plane where the ball will be positioned. The horizontal angular aperture of the camera's field of view is $\frac{\pi}{3}$ radians (approx. 60^{o}), and we delimited the separation from the camera to the ball between distances of $0.5m$ and $2m$. The projection on the ground of the Aibo´s center of mass in its nominal initial configuration of figure 3.15 is defined as the origin of the world reference system. The floor in front of the robot was divided in 18 sample positions, as illustrated in figure 3.16, to evaluate the effect of the uncertainty and variability on the ball position, each position was represented by a circle of $13cm$ of radius and was divided in 32 points evenly distributed, with $2cm$ separation between points, along each axis. In figure 3.17 we show the distribution of the points inside each circle. We have a total of 576 points into the vision range of the Aibo robot where we can place the ball to perform the Visual Servoing experiments.

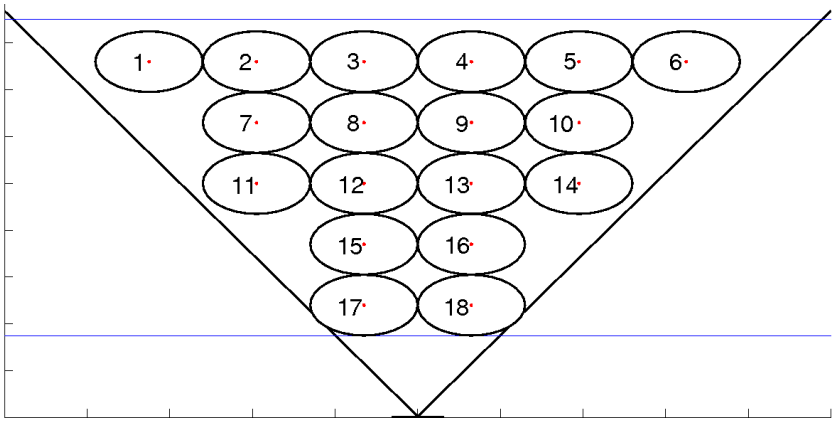


Figure 3.16: Floor space division. Aibo's position is in the lower vertex of the triangle.

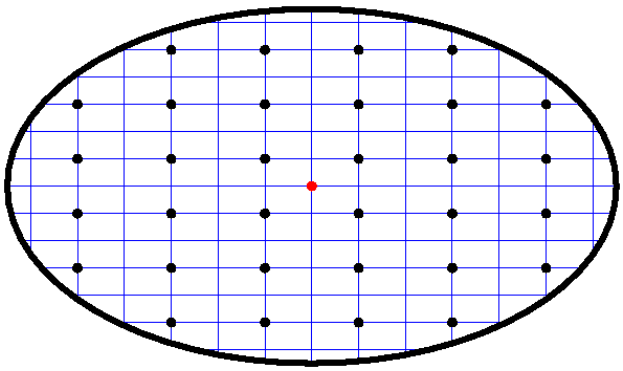


Figure 3.17: Uniform sampling in an uncertainty region around a ball position sampling point.

We can estimate the distance of the ball to the camera image plane taking into account the actual diameter of the ball and its diameter in the image plane (equation 3.38) and use it to estimate the ball position in the camera reference system (equation 3.39). Errors in the estimated distance produce errors in the estimated position of the ball in the camera reference system.

Experimental results

The range of image plane coordinates is $[-1, 1]$ for both axes. After testing all the possible positions for the ball, the average norm of the final error in the image plane is 0.0783 with a variance of 0.0027. In table 3.3 the average and variance of the ball center final error for each position and uncertainty circle is presented.

	Error norm		u-axis error		v-axis error	
Circle	Average	Variance	Average	Variance	Average	Error
1	0.0821	0.0024	-0.0372	0.0030	0.0600	0.0012
2	0.0732	0.0007	-0.0063	0.0012	0.0505	0.0023
3	0.0642	0.0011	-0.0070	0.0009	0.0478	0.0020
4	0.0886	0.0097	-0.0235	0.0079	0.0699	0.0043
5	0.0757	0.0006	-0.0028	0.0003	0.0672	0.0015
6	0.0874	0.0062	0.0111	0.0069	0.0696	0.0021
7	0.0704	0.0008	-0.0125	0.0010	0.0515	0.0020
8	0.0791	0.0016	-0.0206	0.0013	0.0624	0.0023
9	0.0935	0.0016	-0.0027	0.0006	0.0866	0.0023
10	0.0860	0.0002	-0.0063	0.0007	0.0708	0.0020
11	0.0766	0.0010	-0.0238	0.0006	0.0661	0.0014
12	0.0868	0.0056	-0.0291	0.0065	0.0659	0.0015
13	0.0698	0.0016	-0.0001	0.0011	0.0539	0.0025
14	0.0737	0.0008	-0.0019	0.0010	0.0565	0.0021
15	0.0740	0.0097	0.0048	0.0071	0.0509	0.0056
16	0.0880	0.0004	0.0047	0.0005	0.0837	0.0007
17	0.0692	0.0038	-0.0161	0.0011	0.0530	0.0045
18	0.0665	0.0006	0.0344	0.0007	0.0503	0.0007
Total	0.0783	0.0027	-0.0083	0.0026	0.0623	0.0024

Table 3.3: Average Visual Servoing final error at each uncertainty circle.

Figure 3.18 plots the error norm at the end of the Visual Servoing process versus the Euclidean distance, in the 3D world reference system, at which the ball was placed from the camera image plane. Notice that the magnitude of the error does not show any trend related to the distance to the camera, therefore it can be said that the Visual Servoing performance is nearly invariant relative to the distance of the target object to the camera. The vertical structures in the plot correspond to a collection of experiments where the ball was perceived at similar distances. Those structures show an uniform distribution of the final error wich are pretty similar among them. Therefore, the distribution of the final error can also be assumed invariant to the distance of the ball to the robot.

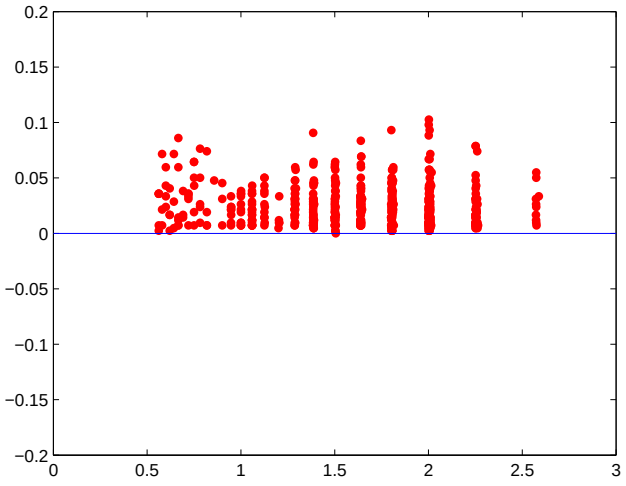
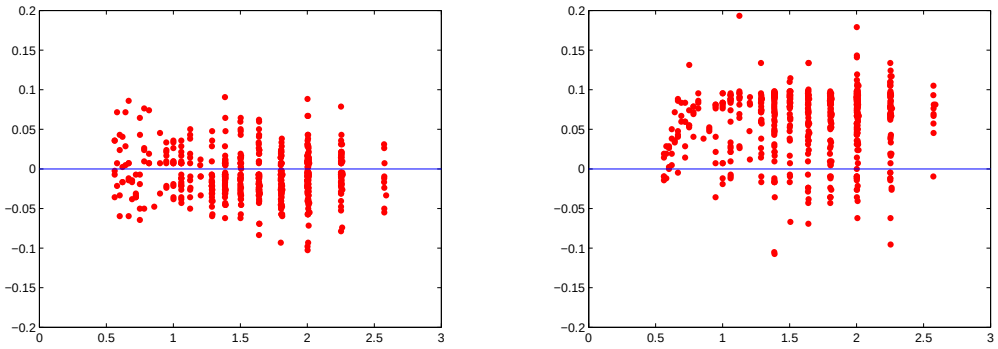


Figure 3.18: Final Visual Servoing error norm distribution versus distance to the ball in the 3D world reference system.

In order to refine this observation, the distribution of the final Visual Servoing error in the u-axis and the v-axis versus the perceived distance of the ball is presented in figure 3.19. The average value for the u-axis is -0.0166 with a variance of 0.0102, while for the v-axis the average value is 0.1246 with a variance of 0.0095. Therefore we appreciate a significative bias of the error that makes it greater in the v-axis. However the error distribution remains invariant to the perceived distance of the ball.



(a) u-axis (b) v-axis

Figure 3.19: Final distribution of the components of the Visual Servoing error versus distance to the ball.

In figure 3.20 we plot the norm of the final error versus the initial distance of the ball center to the image plane center. The figure does not show the column structures as in figure 3.18. We have a nearly uniform distribution of the initial distances. Again the error is invariant to the initial distance in the image plane.

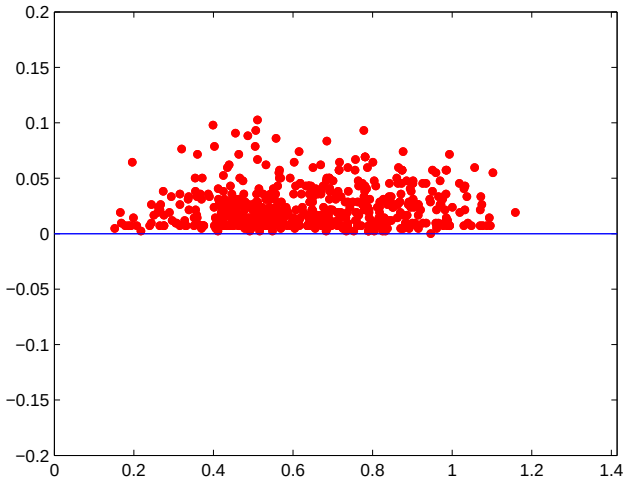
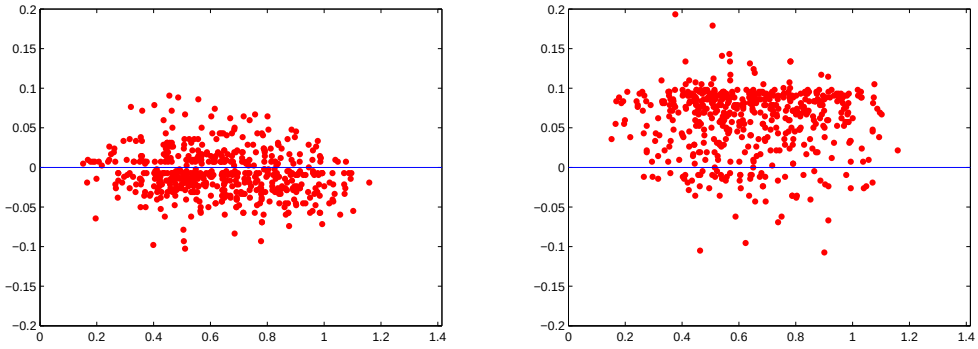


Figure 3.20: Final error norm distribution over initial distance in image plane.

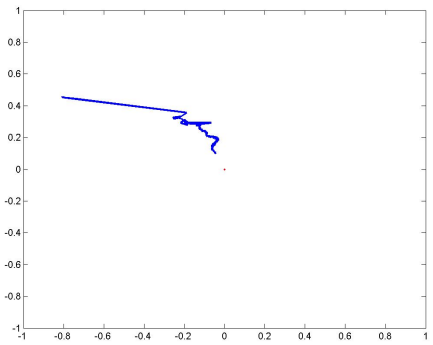
As before, we plot the final error in the u-axis and v-axis versus the initial perceived distance of the ball center to the image plane center in figure 3.21. Again we find that the distribution of the eror is invariant to initial distance in the image plane.



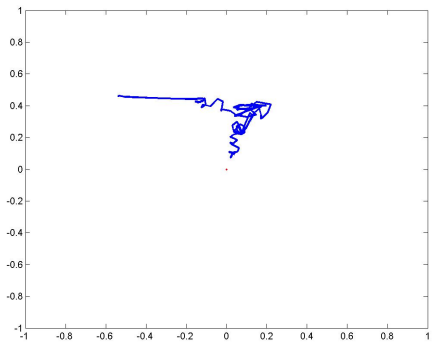
(a) u-axis (b) v-axis

Figure 3.21: Final error distribution over initial distance in image plane.

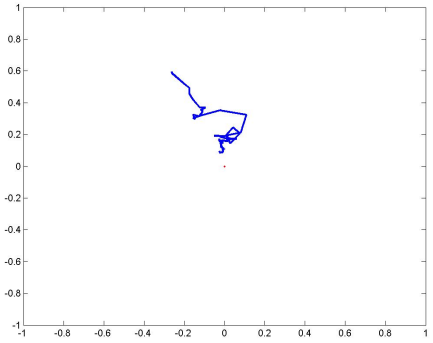
In figures 3.22, 3.23 and 3.24, we show some sample trajectories of the ball center in the image plane along the Visual Servoing process. Most of them show a fairly smooth convergence to the image center



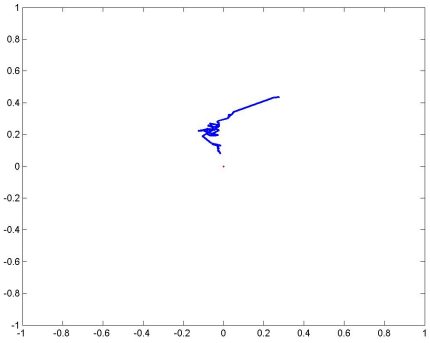
(a)



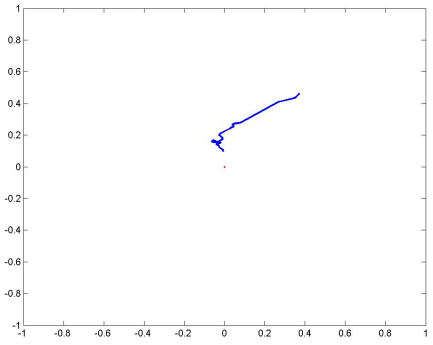
(b)



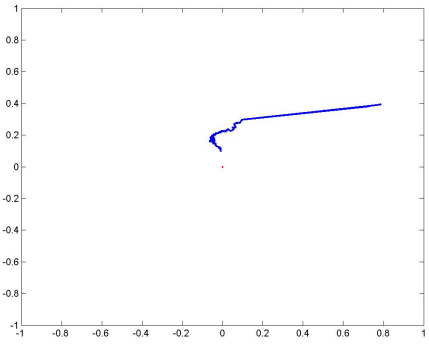
(c)



(d)



(e)



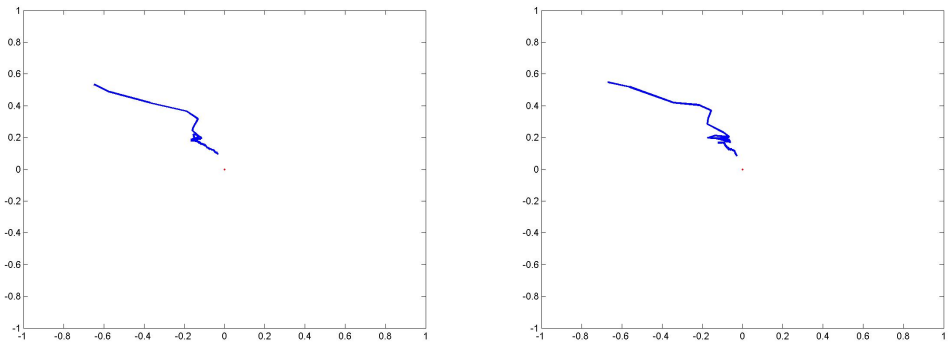
(f)

Figure 3.22: Sample trajectories of the ball center in the image with the ball placed in some position inside the uncertainty circle (positions 1 to 6).

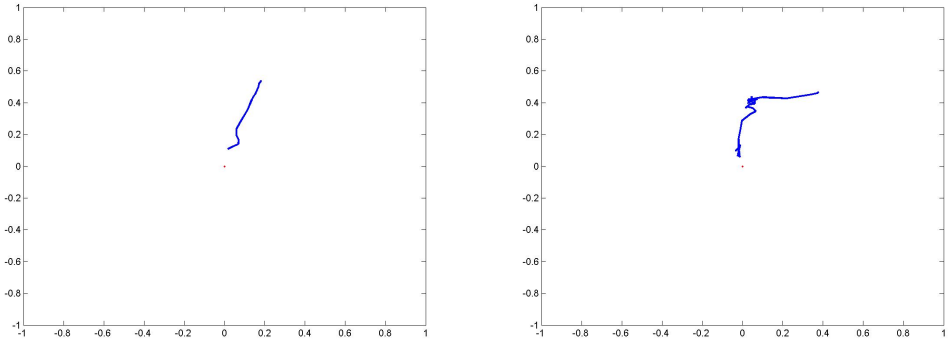
In order to get some information about the smoothness of the Visual Servoing trajectories we have computed the average spatial derivative of the trajectories, starting from the sampling points in each uncertainty circle associated with each ground plane sampling position. These values are shown in table 3.4. We can appreciate that some of the starting ball positions produce systematically smooth trajectories, such as position 17 and 18, while others, such as positions 6 and 10, produce more jumpy trajectories. The reasons for that behaviour lie in the uneven distribution of the servo-motors response power and control resolution on the Aibo's articulations, as well as the image segmentation problems.

	Norm		u-axis error		v-axis error	
Circle	Average	Variance	Average	Variance	Average	Error
1	0.0065	0.0016	0.0065	0.0016	0.0010	0,00018
2	0.0065	0.0003	0.0065	0.0003	0.0057	0,00015
3	0.0059	0.0002	0.0059	0.0002	0.0051	0,00011
4	0.0065	0.0009	0.0065	0.0009	0.0048	0,00051
5	0.0052	0.0005	0.0052	0.0005	0.0027	0,00006
6	0.0089	0.0027	0.0089	0.0027	0.0045	0,00032
7	0.0061	0.0004	0.0061	0.0004	0.0036	0,00009
8	0.0050	0.0002	0.0050	0.0002	0.0032	0,00007
9	0.0059	0.0003	0.0059	0.0003	0.0040	0,00013
10	0.0090	0.0006	0.0090	0.0006	0.0041	0,00008
11	0.0050	0.0005	0.0050	0.0005	0.0032	0,00007
12	0.0029	0.0002	0.0029	0.0002	0.0022	0,00006
13	0.0051	0.0007	0.0051	0.0007	0.0039	0,00021
14	0.0066	0.0005	0.0066	0.0005	0.0036	0,00008
15	0.0051	0.0002	0.0051	0.0002	0.0038	0,00007
16	0.0030	0.0002	0.0030	0.0002	0.0038	0,00004
17	0.0029	0.0002	0.0029	0.0002	0.0023	0,00006
18	0.0011	0.0000	0.0011	0.0000	0.0036	0,00000
Total	0.0052	0.00054	0.0052	0.00054	0.0036	0.00013

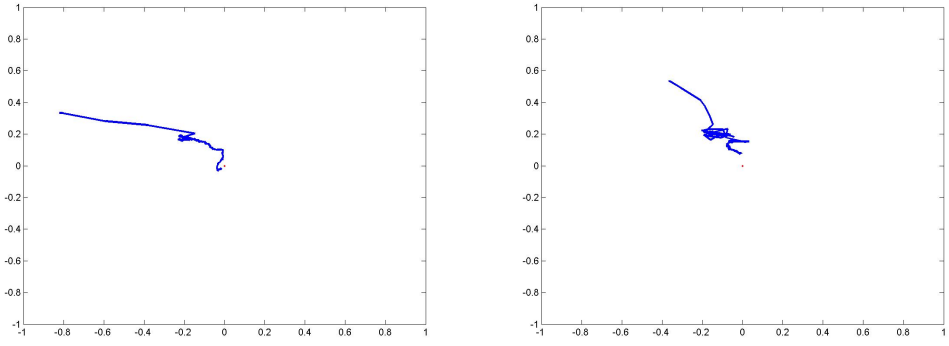
Table 3.4: Trajectory error variations at each uncertainty circle.



(a) (b)

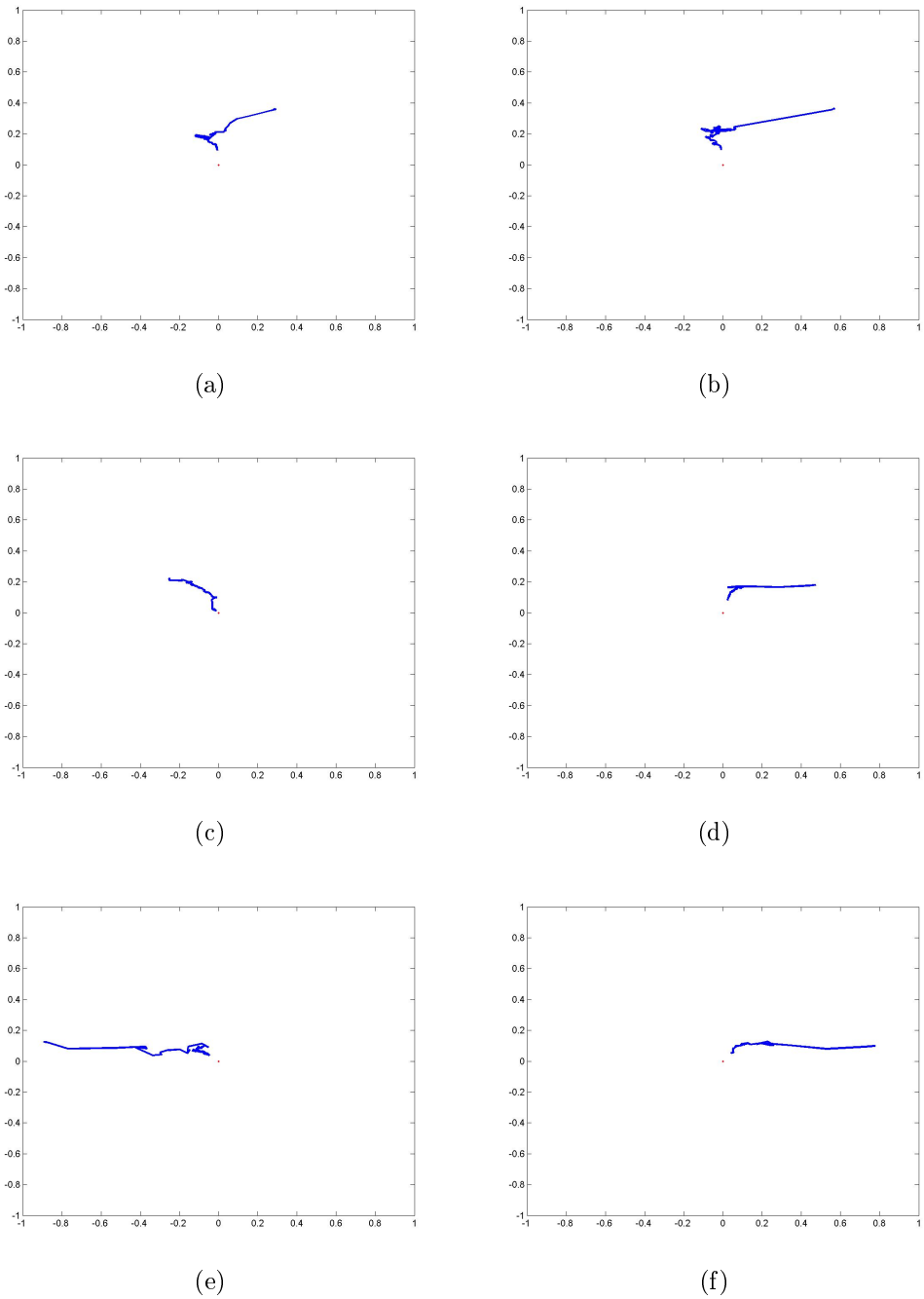


(c) (d)



(e) (f)

Figure 3.23: Sample trajectories of the ball center in the image with the ball placed in some position inside the uncertainty circle (positions 7 to 12).



(a) (b)

(c) (d)

(e) (f)

Figure 3.24: Sample trajectories of the ball center in the image with the ball placed in some position inside the uncertainty circle (positions 13 to 18).

3.4.4.2 Visual tracking for a sequence of ball positions

In the second experiment, we defined several sequences of positions of the ball, each one consisting of four positions. The Aibo performed the Visual Servoing chaining the ending robot configurations after each Visual Servoing process. The ball was static while the Aibo was performing each Visual Servoing process. The aim of this experiment is to study the degradation of performance due to the accumulation of errors.

We have defined horizontal trajectories respect to the initial robot configuration, moving the ball from side to side. Figure 3.25 shows four horizontal sequence directions on the ground reference system. Figure 3.26 shows some example trajectories obtained in this experiment. In the plots, the red, green, blue and black trakectories correspond to the Visual Servoing trajectories performed by the Aibo after each of the four ball positions. It can be appreciated that the robot response is quite smooth for the ensuing positions after having performed the Visual Servoing for the first one, even if the initial position was a "difficult" one.

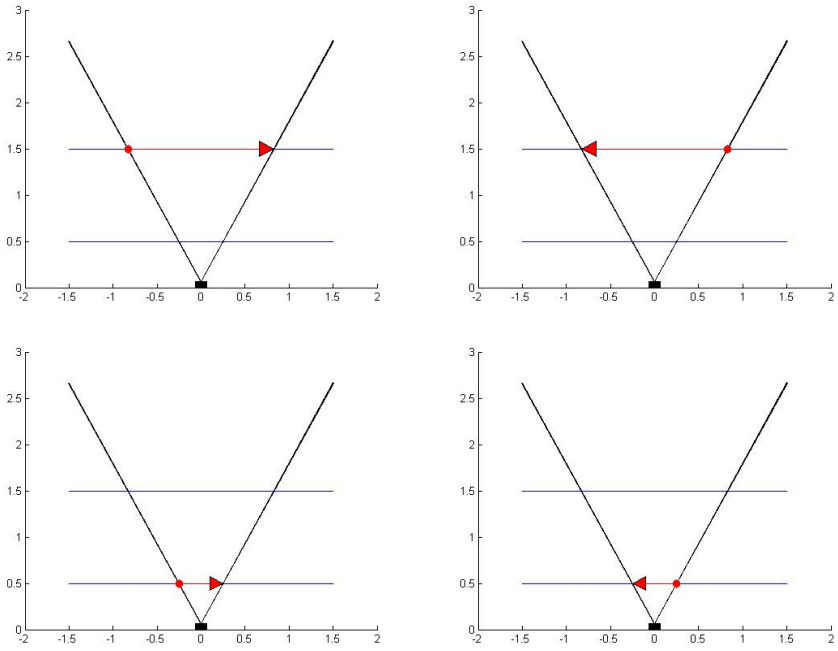


Figure 3.25: Horizontal movements of the ball

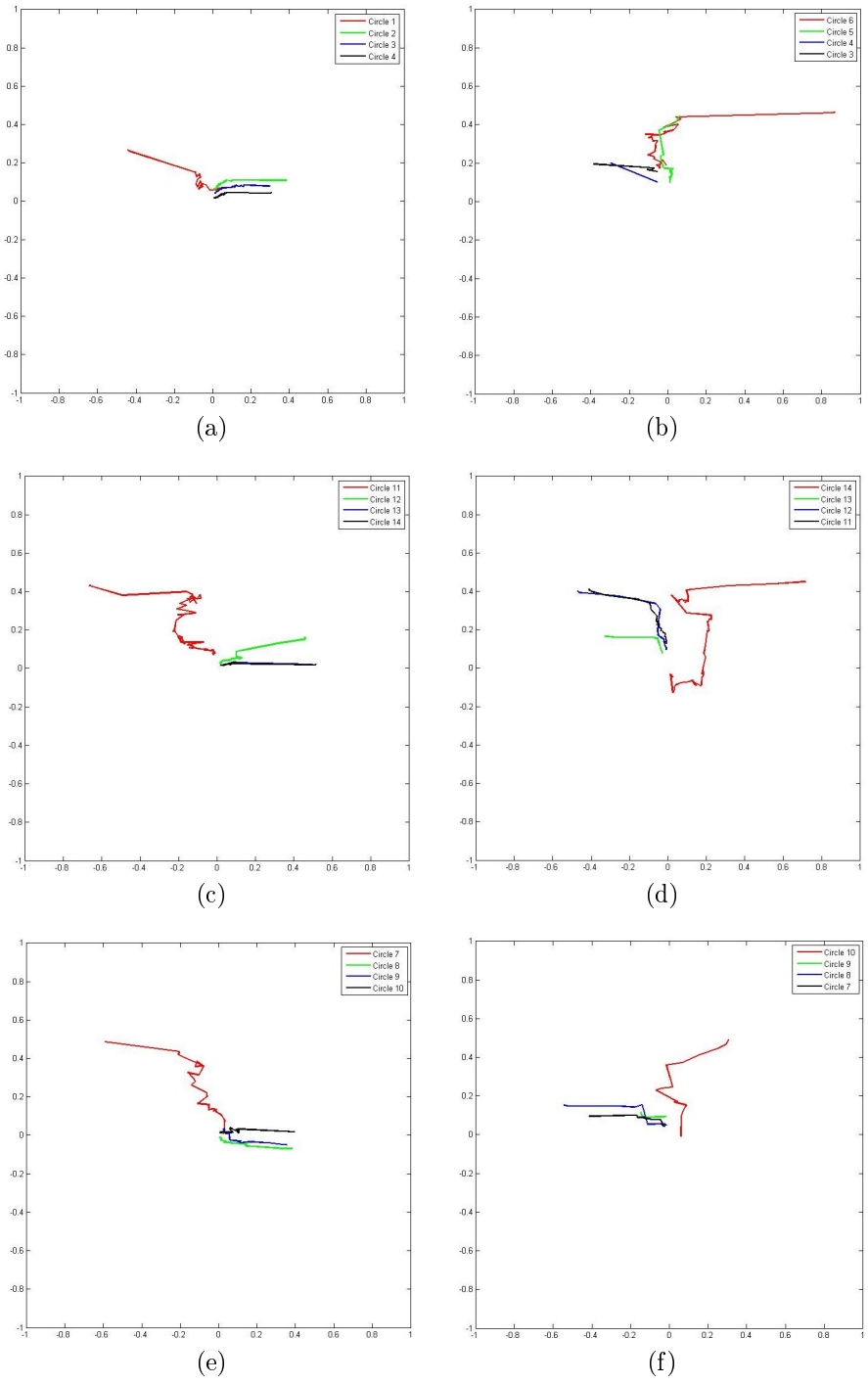


Figure 3.26: Trajectories for a moving ball.

3.5 Conclusions

We have developed the Visual Servoing for the whole set of degrees of freedom of the Aibo ERS-7 following a principled approach. From the geometrical description of the robot we have constructed the full Jacobian matrix that linearizes the functional dependence of the image plane viewed by the robot camera on the robot degrees of freedom. The seudoinverse of this Jacobian matrix provide the desired controls. The blind application of this control strategy may lead the robot to unstable or unfeasible configurations for a standing pose. Therefore, we evaluate an stability condition of the robot configuration. When stability is compromised we restrict the Visual Servoing to the head. The actual implementation in the Aibo ERS-7 shows that the approach performs in real time when the seudoinverse is computed in the on-board processor of the robot. The real life experiments under controlled conditions have shown that the approach is highly robust to positioning of the ball in the field of view of the robot, it performs very fast and with very low final error, independently of the distance of the ball to the camera plane. As the main sources of convergence problems we have identified the following ones: (1) the linear nature of the approach, (2) the control resolution of the physical servo-motors and (3) the problems in the image segmentation.

In the following web link, inside of the Computational Intelligent Group web page, we present a sample video of the experiment `http://www.ehu.es/ccwintco/uploads/0/0a/AiboERS7.mp4`.

Chapter 4

Control of a Multi-robot Hose System

This chapter is devoted to the control of a Multi-Component Robotic System (MCRS) performing the transportation of a hose, aiming to its Visual Servoing, although the achievement of this goal is an on-going research effort that goes beyond the limits of this PhD work. This kind of systems fall in the class of Linked MCRS [15]. In this chapter we first review the motivations for this work in section 4.1, revisiting some of the long term objectives that the research line may pursue. In section 4.2 we describe the geometric and dynamic model of the hose based on the Dynamic Splines modeling approach. In section 4.3 we develop formally the control of the individual robots in order to obtain the desired configuration of the whole system. In section 4.4 we present the simulation of the MCRS Hose system which has been used to test some system properties and to explore the potential behavior of a physical realization. In section 4.5 we report on a real life experiment that, although simplified, projects some light on the difficulties that more extensive efforts will encounter. Finally, section 4.6 gives some conclusions and directions for further work.

4.1 Motivation and objectives

Nowadays robotic systems are facing the challenge of working in very unstructured environments, such as shipyards or construction sites. In these environments, the tasks are non repetitive, the working conditions are dif-

ficult to be modeled or predicted, and the size of the spaces may be huge. Moreover, there are complex tasks that a single robot can not accomplish but that could be achieved by a team of robots. In these environments, a common task is the displacement of some kind of flexible hose. It can be a water hose or a power line, or other. We are interested here in the design of a control architecture for a MCRS dealing with this problem. A collection of cooperating robots attached to the hose must be able to displace it to a desired configuration. The whole system, including the hose, is a paradigmatic example of the class of Linked MCRS [15].

We have identified the following sub-problems:

- Modeling a flexible elongated object, that acts as a passive link between the robots.
- Centralized and/or distributed sensing to obtain information of the environment and/or of the configuration of the system including the robots and the hose.
- Modeling and computing the inverse kinematics of the whole system for its simulation and for the derivation of the control commands for the robots.
- Development of highly adaptive control via high level cognitive mechanisms.

Here we focus on the hose modeling and the generation of control strategies for a collection of autonomous robots attached to it.

Our starting point is modeling the hose geometry taking into account physical models for the internal hose dynamics. The idea is to investigate the area of uni-dimensional object modeling in order to obtain an appropriate representation for the hose. The simulation of the hose-robots system is required in order to design and evaluate control strategies that allow the transport of the hose under different environment conditions, or control strategies, i.e. following a given trajectory for the leader robot. As said before, our hose model is based on the theory of Dynamic Splines. The hose control problem is stated as the problem of reaching a desired configuration of the spline control points from an initial configuration. One of the sub-problems that we studied in depth is the transport of the hose along a trajectory defined for a leader robot. Another is the development of a Vision Servoing approach for the interactive control of the system.

4.2 Hose Model

Modeling uni-dimensional objects has great application for the representation of wires in industry and medicine. Some models use as basic formalism differential equations [49], rigid body chains [26] and spring-mass systems [22]. The combination of spline geometrical modeling and physical dynamical models was introduced by [57]. They allow a continuous definition of uni-dimensional objects. An inconvenient of the spline model is that, since they are based exclusively on the control points of the spline, they are not suitable for representing the hose torsion. The work of [76] has improved the spline representation by combining the splines modeling with the Cosserat rod theory, allowing to model the twisting of the hose. This new approach, known as Geometrically Exact Dynamic Splines (GEDS), represents the control points of the splines by the three Cartesian coordinates plus a fourth coordinate representing the twisting state of the hose.

The Cosserat rod theory [65, 1] is usually used in modeling uni-dimensional objects because it permits to model its physical behavior. In Cosserat rod theory an uni-dimensional object is described by a curve $\mathbf{r}(u)$ and a coordinate frame of director vectors $[\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3](u)$ attached to each point of the curve. The parameter u goes from one end of the curve, for $u = 0$, to the other for $u = L$, being L the length of the hose. The curve and the director vectors are joined into a coordinate frame $E(u) = [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{r}](u)$. A graphic representation of the hose by the curve and the frame director vectors is shown in figure 4.1.

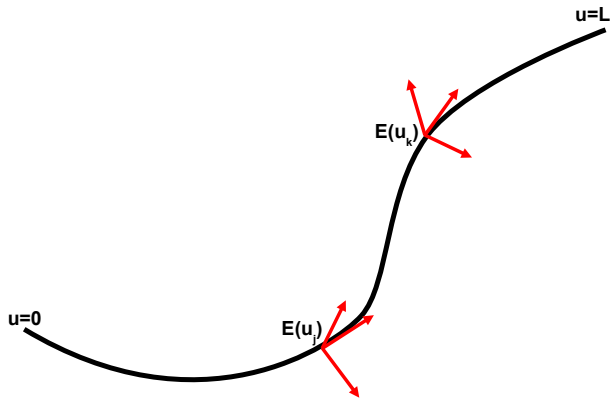


Figure 4.1: Cosserat rod model of a hose.

The GEDS describe the uni-dimensional object by a spline model taking into account the Cosserat rod approach in order to model the twisting behavior of the hose. A spline is a piecewise polynomial function. See figure 4.2 for an illustration. Splines define a curve by means of a collection of Control Points, which define a function that allows to compute the whole curve.

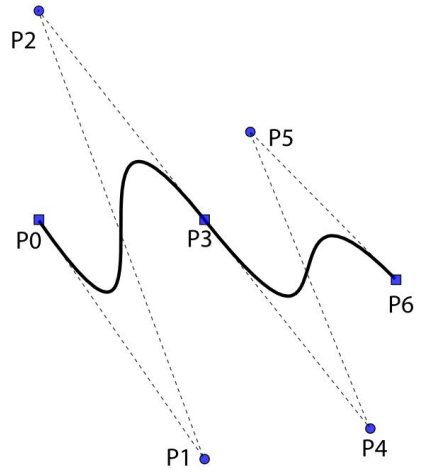


Figure 4.2: Cubic spline.

The spline expression for a curve $\mathbf{q}(u)$ is a linear combination of control points $\mathbf{p}_i$ where the linear coefficients are the polynomials $N_i(u)$ which depend on the parameter u defined in $[0, 1)$.

There are several kinds of polynomials used in the literature of geometric modeling, and depending of the type selected the curve specification follows a specific pattern. In the following equation 4.1 the spline definition is presented:

$$\mathbf{q}(u) = \sum_{i=0}^{n} N_i(u) \cdot \mathbf{p}_i, \tag{4.1}$$

where $N_i(u)$ is the polynomial associated to the control point $\mathbf{p}_i$, and $\mathbf{q}(u)$ is the point of the curve at the parameter value u. It is possible to travel over the curve by varying the value of parameter u, starting at one end for $u = 0$ and finishing at the other end for $u = 1$.

In our work we have used B-spline for modeling the hose because it is a spline function that has minimal support with respect to a given degree, smoothness, and domain partition. Moreover, a fundamental theorem states

that every spline function of a given degree, smoothness, and domain partition, can be represented as a linear combination of B-splines of that same degree and smoothness, and over that same partition [3]. When designing a B-spline curve, we only need a set of control points, a set of knots and a set of coefficients, one for each control point, so that all curve segments are joined together satisfying certain continuity condition.

Given $n+1$ control points $\{\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_n\}$ and a knots vector $\mathbf{U} = \{u_0, u_1, \ldots, u_m\}$, the B-spline curve of degree p defined by these control points and knots vector $\mathbf{U}$ is:

$$\mathbf{q}(u) = \sum_{i=0}^{n} N_{i,p}(u) \cdot \mathbf{p}_i, \tag{4.2}$$

where $N_{i,p}(u)$ are B-spline basis functions of degree p.

The basis functions are calculated by the Cox de Boor's algorithm:

$$\begin{aligned} N_{i,0}(u) &= \begin{cases} 1 & u_i \leq u < u_{i+1} \\ 0 & c.c. \end{cases} \\ N_{i,p}(u) &= \frac{u - u_i}{u_{i+p} - u_i} \cdot N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} \cdot N_{i+1,p-1}(u) \end{aligned}$$

Because the control points of the curve will vary in time, we rewrite equation 4.2 in terms of the time parameter t:

$$\mathbf{q}(u,t) = \sum_{i=0}^{n} N_{i,p}(u) \cdot \mathbf{p}_i(t). \tag{4.3}$$

This extended model receives the name of *Dynamic splines.* From now on we will use cubic B-splines curves ($p = 3$) and the notation for the cubic basis functions will be:

$$N_{i,p}(u) = N_{i,3}(u) \equiv N_i(u).$$

When modeling a hose, we assume that it has a constant sectional diameter, and that the transverse sections are not deformed in any way. If we do not take into account the hose internal dynamics, a spline passing through all the transverse section centers suffices to define the hose, as can be appreciated in figure 4.3. If we want to take into account the hose internal dynamics, we need also to include the hose twisting at each point given by the rotation of the transverse section around the axis normal to its center point, in order to compute the hose potential energy induced forces. In the GEDS

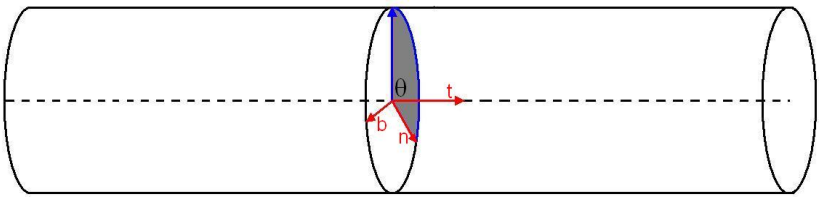


Figure 4.3: Hose section.

model, the hose follows the Cosserat rod approach and then it is described by the collection of transverse sections. To characterize them it suffices to have the curve given by the transverse section centers $c = (x, y, z)$, and the orientation of each transverse section θ. This description is summarized by the following notation: $\mathbf{q} = (\mathbf{c}, \theta) = (x, y, z, \theta)$. In figure 4.3, the relation between the Cosserat rod director vectors and the twisting angle θ is shown, where vector $\mathbf{t}$ represents the tangent to the curve at point $\mathbf{c}$, and vectors $\mathbf{n}$ and $\mathbf{b}$ determine the angle θ of the transverse section at point $\mathbf{c}$.

From the Cosserat representation and applying the Lagrange equation (equation 4.4) we have the mathematical relation between the potential energy U, the Kinetic energy T and the generalized external forces $\mathbf{F}$.

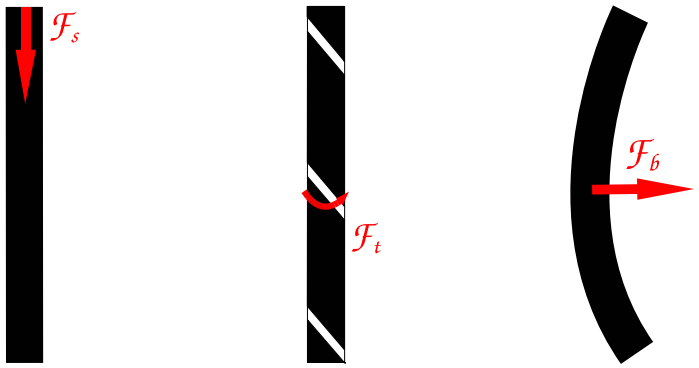
$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\mathbf{p}}_i}\right) = \mathbf{F}_i - \frac{\partial U}{\partial \mathbf{p}_i}. \tag{4.4}$$

The kinetic energy is the motion energy, while the potential energy is the energy stored because of the hose position. $\mathbf{F} = \{\mathbf{F}_0, \mathbf{F}_1, \ldots, \mathbf{F}_n\}$ is the model of the external forces acting on the hose spline model control points. It is usually assumed that mass and stress are homogeneously distributed among the $n + 1$ degrees of freedom of the hose spline control model.

4.2.1 Potential Energy

It is necessary to determine the forces that will be generated on the hose as a consequence of its potential energy due to its physical configuration.

In figure 4.4 we can appreciate the forces and torques $\mathbf{F}_U = (\mathbf{F}_s, \mathbf{F}_t, \mathbf{F}_b)^T$ that deform the hose because of its potential energy. The stretching force, $\mathbf{F}_s$, is the force along the normal to the hose transverse section and its application results in its lengthening. The tension torque, $\mathbf{F}_t$, makes the transverse section to rotate around the center of the section. The bending torque, $\mathbf{F}_b$,



(a) Stretching force. (b) Twisting force. (c) Bending force.

Figure 4.4: Forces induced by the potential energy of the hose.

modifies the orientation of the transverse section. The forces acting on the transverse section plane are neglected, because of the Kirchhoff assumption that the transverse sections are rigid and that only the hose curvature may be distorted.

In mechanics and physics the Hooke's law provides an approximation for linear-elastic materials. This law establishes that the extension of a spring is in direct proportion to the load applied to it. Summarizing, the Hooke's law for a spring-mass system establishes:

$$F = -kx, \tag{4.5}$$

where x is the displacement of the spring due to the load applied to it, k is the spring constant and F the restoring force experimented by the spring due to its material properties. In general the Hooke's law is applied to elastic materials because their behavior is similar to the spring as its molecules return to the initial state of stable equilibrium, quickly regaining the object its original shape after a force has been applied.

Let us denote the length of the hose as L, and the area of the transverse section as A, then the hose length extension is linearly proportional to the deformation resistance of the hose:

$$\triangle L = \frac{F}{EA}L, \tag{4.6}$$

where E is the modulus of elasticity, which is the mathematical description

for the hose resistance to be deformed when a force is applied to it.

Isolating the value of F in equation 4.6 we have:

$$F = EA\frac{\triangle L}{L}. \tag{4.7}$$

Defining ϵ as the deformation of the hose relative to the transverse area, $\epsilon = A\frac{\triangle L}{L}$, we can rewrite 4.7 as:

$$F = E\epsilon, \tag{4.8}$$

which is a version of the Hooke's law for an elastic uni-dimensional object.

When a small deformation is considered for a relative big radius of the hose length in comparison with the radius of the transverse section, it is said that the hose is in a linear elasticity dynamic regime, and then the force equation 4.8 may be applied for each of the stretching, twisting and bending forces. The matricial version for the stretching, twisting and bending forces is:

$$F_U = H\epsilon = \begin{pmatrix} E_s & 0 & 0 \\ 0 & E_t & 0 \\ 0 & 0 & E_b \end{pmatrix} \cdot \epsilon. \tag{4.9}$$

The deformation vector, $\epsilon = (\epsilon_s, \epsilon_t, \epsilon_b)^T$, is composed of the stretching deformation ϵ_s, the twisting deformation ϵ_t and the bending deformation ϵ_b. The Hooke matrix, H, is composed of the stretching rigidity E_s, the twisting rigidity E_t and the bending rigidity E_b.

Maintaining the spring-mass system analogy, the potential energy U is defined as $U = \frac{1}{2}kx^2$, that in the case of the hose is defined by the following integration from $u = 0$ up to $u = L$:

$$U = \frac{1}{2}\int_0^L \epsilon^T \mathbf{F}_U du. \tag{4.10}$$

Using the definition of $\mathbf{F}_U$ from equation 4.9 in equation 4.10 we have:

$$U = \frac{1}{2}\int_0^L \epsilon^T H\epsilon du$$

Note that this model is appropriated for a hose that in rest configuration is stiffed and not twisted or bended, but for a cable as a telephone cord or

a spring the rest configuration of the hose is different to zero, so ϵ should be replaced by $(\epsilon - \epsilon_0)$, being ϵ_0 the rest strain.

4.2.2 Kinetic energy

The kinetic energy T is composed of the translation energy T_t and the rotation energy T_r.

$$T = T_t + T_r. \tag{4.11}$$

The kinetic energy is given by:

$$T_t = \frac{1}{2}\mu A \int_0^L \dot{\mathbf{q}}^2 du, \tag{4.12}$$

$$T_r = \frac{1}{2}\mu \int_0^L \mathbf{\Omega}^T I \mathbf{\Omega} du, \tag{4.13}$$

where A is the area of the transversal section,$\mathbf{\Omega}$ is the angular velocity, μ is the linear density and I is the polar momentum of inertia.

A simplified version of the kinetic energy expression is given by defining the inertial matrix J, which is invariant over all the hose points because of the assumption of constant hose diameter.

$$J = \begin{pmatrix} \mu & 0 & 0 & 0 \\ 0 & \mu & 0 & 0 \\ 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & I \end{pmatrix}.$$

The kinetic energy of the hose T is then defined by:

$$T = \frac{1}{2}\int_0^L \frac{d\mathbf{q}^T}{dt} J \frac{d\mathbf{q}}{dt} du. \tag{4.14}$$

4.2.3 Dynamic model

The kinetic energy model takes into account the translational and rotational motions of the hose, therefore, we can determine from it the acceleration of every hose point, by deriving equation 4.14. The left hand term of equation 4.4 becomes:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\mathbf{p}}_i}\right) = \frac{1}{2}\int_0^L \frac{d}{dt}\frac{\partial\left(\dot{\mathbf{q}}^T J \dot{\mathbf{q}}\right)}{\partial \dot{\mathbf{p}}_i} du. \tag{4.15}$$

Next, we consider the derivative of the potential energy relative to a generalized coordinate:

$$\frac{\partial U}{\partial \mathbf{p}_i} = \frac{1}{2}\int_0^L \frac{\partial \epsilon^T H \epsilon}{\partial \mathbf{p}_i} du. \tag{4.16}$$

The aim of the physical modeling is to determine the accelerations of the hose, in terms of its geometrical model, therefore the accelerations are obtained in the GEDS model substituting $\mathbf{q}$ in the expression of equation 4.15 by the right side of equation 4.3:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\mathbf{p}}_i} = \sum_{j=0}^{n} J \frac{d^2\mathbf{p}_j}{dt^2}\int_0^L (N_i(u)N_j(u))du \tag{4.17}$$

Defining:

$$\mathbf{M}_{ij} = J\int_0^L (N_i(u)N_j(u))du$$

and

$$\mathbf{A} = \left[\frac{d^2\mathbf{p}_j}{dt^2}\right],$$

The Lagrange equation (equation 4.4) becomes:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\mathbf{p}}_i} = \sum_{j=0}^{n} \mathbf{M}_{i,j}\mathbf{A}_j. \tag{4.18}$$

Using equations 4.18 and 4.16 the Lagrange equation is written in a matrix form:

$$\mathbf{MA} = \mathbf{F} + \mathbf{P}, \tag{4.19}$$

where $\mathbf{P} = \left[\frac{\partial U}{\partial \mathbf{p}_i}\right]$.

4.2.4 MCRS Hose configuration

The last step in the hose modeling is to define the configuration of the system composed by the hose and the robots, taking into account the control points and knots of the B-spline representation, and the u parameter values that give the position of the robots along the hose. A configuration h of the hose-robots system is defined as:

$$h = \{\mathbf{p}, \mathbf{U}, \mathbf{U_r}\}, \tag{4.20}$$

where:

- $\mathbf{p}$ is the control point vector of the hose B-spline model.
- $\mathbf{U}$ is the vector of knots in the B-spline model.
- $\mathbf{U_r} \subset \mathbf{U}$ is the robot knot vector.

The robot knot vector $\mathbf{U_r}$ contains the values of the parameter u where the robots are attached to the hose. If we denote u_{r_i} the value for the i-th robot in $\mathbf{U_r}$, then the position of the spline at u_{r_i} is the spatial position of the i-th robot $\mathbf{r}_i$:

$$\mathbf{q}(u_{r_i}) = \sum_{i=0}^{n} N_i(u_{r_i}) \cdot \mathbf{p}_i = \mathbf{r}_i. \tag{4.21}$$

The information we have about the hose is a sequence τ of sampling points of the hose center curve, containing the Cartesian position of every point (x, y, z) and its torsion angle θ. Then, we construct the initial hose configuration h_0 by an interpolation method that generates the control points of the B-spline cubic curve that interpolates this points. The method we used is the Interpolating Forward Backward Algorithm (IFBA) for clamped B-spline cubic curves, and a description of this method is presented in Appendix A.

More precisely, we make an uniform sampling of the hose center curve to obtain the sequence of points τ in order to get an uniform B-spline interpolant. This distribution of the sampling points optimizes the performance of the IFBA, avoiding the occurrence of spurious peaks, protuberances and loops. The sampling is done taking into account the number of knots we want to use, depending on the relation between precision and computing time that we desire.

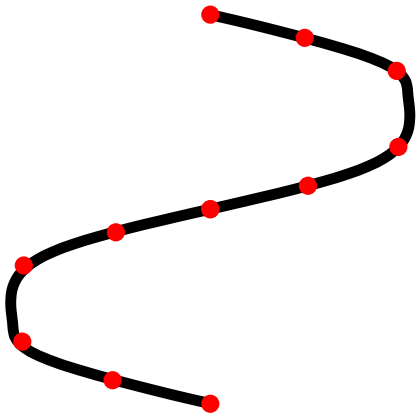


Figure 4.5: Uniform selection of the interpolating points.

The uniform selection of the interpolating points τ is obtained by dividing the hose length into n segments, being $n+1$ the number of control points, and choosing for each division point the hose point closest to it. Figure 4.5 shows the hose in black and the selected interpolating points in red, for a number of control points $n = 10$.

This reconstruction of the hose configuration from a sequence of points may be useful when we have a vision system that provides us with a sequence of points corresponding the segmentation of the hose in the image, and the positions of the robots contact points.

4.3 MCRS Hose control

We define two basic kinds of hose positioning tasks accomplished through the positioning of several autonomous robots $\mathbf{r} = \{\mathbf{r}_1, \ldots, \mathbf{r}_l\}$ attached to the hose. The first task is to bring the hose from an initial configuration to a final configuration determining at each instant the velocities that the robots must experiment. This work is developed in the following section 4.3.1. The second task is stated as: given a trajectory for the leader robot (the robot at the front end of the hose corresponding to $u = 0$) the remaining of the robots must follow it in order to accomplish the transport of the hose. This work is developed in section 4.3.2.

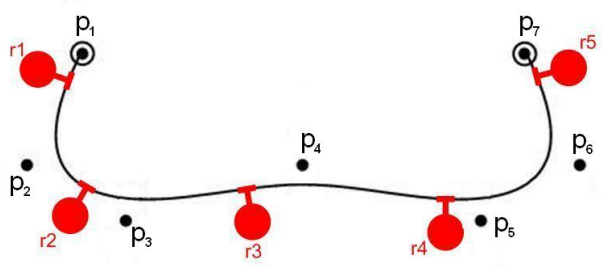


Figure 4.6: Control points $\mathbf{p}_i$ of the spline and positions of the robots $\mathbf{r}_i$.

4.3.1 Hose control for the transition among configurations

The first of the proposed tasks is specified by giving the initial and the final hose configurations, as well as the initial robot positions. In figure 4.6 we show a typical configuration of the system, with a hose described by parametric cubic splines with control points $\mathbf{p}_i$ and a collection of robots $\mathbf{r}_j$ attached to it.

Let it be:

- h_0 the initial hose configuration obtained from a sampling sequence of points as described above.
- h_* the desired hose representation.
- $\mathbf{r}_0 = \{\mathbf{r}_1, \ldots, \mathbf{r}_l\}$ the robot initial positions.

We are interested in obtaining the motion of the attached robots, given by the instantaneous velocities of the hose attachment points $\dot{\mathbf{r}}$, which will move the hose from the initial configuration representation h_0 to a desired configuration h_* starting from an initial configuration of the robots $\mathbf{r}_0$. In order to use the spline model of the hose, we need to obtain a B-spline representation for the hose by interpolating an uniform selection of the given sets of points h_0 and h_* as explained in section 4.2.4. For the purpose of the design of the control process, the hose configurations are well described by the sequence of control points vectors. The control process will be computed as a sequence of transitions from $\mathbf{p}_0$ to $\mathbf{p}_*$.

4.3.1.1 Derivation of the control law

In order to obtain the desired velocities of the hose control points that reduce the distance between their current positions and the desired ones, we define the hose configuration error as the difference between the current control points and the desired one as:

$$e(\mathbf{p}) = (\mathbf{p}_* - \mathbf{p})^2.$$

The error function allows us to define the following simple proportional control law:

$$\dot{\mathbf{p}} = k\,(\mathbf{p}_* - \mathbf{p})\,. \tag{4.22}$$

This expression defines a differential equation on the control points. Solving it we obtain the following trajectory for the control points:

$$\mathbf{p}(t) = \mathbf{p}_0 \cdot e^{-k(t-t_0)} + \mathbf{p}_* \cdot \left[1 - e^{-k(t-t_0)}\right],$$

where t_0 is the time instant at which $\mathbf{p}(t_0) = \mathbf{p}_0$.

Since our system is expressed in terms of accelerations, we have to obtain the accelerations at the control points:

$$\ddot{\mathbf{p}}(t) = \frac{d\dot{\mathbf{p}}(t)}{dt}.$$

Since we have a discrete iterative control process, we define the acceleration in the $k+1$ step in terms of the desired velocity in $k+1$, the current velocity in k and the time increment:

$$\ddot{\mathbf{p}}_{k+1} = \frac{\dot{\mathbf{p}}_{k+1} - \dot{\mathbf{p}}_k}{\triangle t}. \tag{4.23}$$

We need to determine the forces that robots should exert in order to obtain the desired accelerations on the control points.

4.3.1.2 Forces applied by the robots on the hose

Equation 4.19 relates the acceleration at the control points with the internal energy of the hose and the external forces applied to it. Among the external forces $\mathbf{F}$ that act on the control points, we differentiate those resulting from the ones applied by the robots $\mathbf{F}_p$ from other external forces, $\mathbf{F}_e$:

$$\mathbf{F} = \mathbf{F}_p + \mathbf{F}_e. \tag{4.24}$$

So, we rewrite equation 4.19 in order to determine the forces that the robot must exert on the hose attachment points as a function of the desired accelerations on the control points, the external forces on the hose and the energy configuration:

$$\mathbf{F}_p = M\mathbf{A} - \mathbf{F}_e - \mathbf{P}. \tag{4.25}$$

Since the hose dynamics are defined on the control points $\mathbf{p}_i$, a force $\mathbf{f}$ applied on a particular point of the hose is decomposed into the forces $\mathbf{f}_i$ resulting at each spline control point $\mathbf{p}_i$. The partial derivative of a point $\mathbf{q}(u)$ in the curve respect to the control point $\mathbf{p}_i$ is:

$$\frac{d\mathbf{q}(u)}{d\mathbf{p}_i} = N_i(u). \tag{4.26}$$

For a control point $\mathbf{p}_i$, the corresponding force $\mathbf{f}_i$ is computed as:

$$\mathbf{f}_i = \mathbf{f}\frac{\partial \mathbf{q}}{\partial \mathbf{p}_i} = \mathbf{f} \cdot N_i. \tag{4.27}$$

Defining the Jacobian matrix J_{rq} of the robot contact points with the hose as a function of the control points, we have:

$$J_{pr} = \begin{pmatrix} \frac{\partial \mathbf{q}(u_{r_1})}{\partial \mathbf{p}_0} & \cdots & \frac{\partial \mathbf{q}(u_{r_l})}{\partial \mathbf{p}_0} \\ \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{q}(u_{r_1})}{\partial \mathbf{p}_n} & \cdots & \frac{\partial \mathbf{q}(u_{r_l})}{\partial \mathbf{p}_n} \end{pmatrix} = \begin{pmatrix} N_0(u_{r_1}) & \cdots & N_0(u_{r_l}) \\ \vdots & \ddots & \vdots \\ N_n(u_{r_1}) & \cdots & N_n(u_{r_l}) \end{pmatrix}, \tag{4.28}$$

where u_{r_j} is the attachment point of the robot $\mathbf{r}_j$ to the hose.

We use the Jacobian matrix J_{pr}, defined in equation 4.28, to obtain the relation between the applied forces in the robot attaching points $\mathbf{F_r}$ and the resulting forces on the control points $\mathbf{F_p}$:

$$\mathbf{F_p} = J_{pr} \cdot \mathbf{F}_r. \tag{4.29}$$

The previous control law gives us the desired accelerations on the control points, so we can derive the desired forces $\mathbf{F}_p$ that must be resulting on the control points from the robot actions. Therefore we have to determine the

forces $\mathbf{F}_r$ that the robots must apply on their contact points with the hose, by inversion of equation 4.29. But in general, matrix J_{pr} is not invertible.

We consider three possible cases depending of the number of control points n and the number of robots l.

- If $l = n$, then the inverse of J_{pr} exists and we can compute $\mathbf{F}_r = J_{pr}^{-1}\mathbf{F}_p$.
- In the other two cases, when $l \neq n$, the inverse does not exist. Assuming that J_{pr} is full rank, then its pseudo-inverse can be computed to solve the problem by least squares, the solution is:

$$\widehat{\mathbf{F}_r} = J_{pr}^{+}\mathbf{F}_p + (I - J_{pr}^{+}J_{pr})\mathbf{w}_l, \tag{4.30}$$

  where J_{pr}^{+} is a pseudo-inverse matrix of J_{pr} and $\mathbf{w}_l \in R^{4l}$. The solution by least squares allows us to obtain a value for $\widehat{\mathbf{F}_r}$ that minimizes the following norm $\left\|\mathbf{F}_p - J_{pr}\widehat{\mathbf{F}_r}\right\|$. In obtaining the pseudo-inverse we have to take into account two possible cases:

  - First, if $n > l$ there are more control points than robots and then, from the Theorem of the Implicit Function, the control points $\mathbf{p}_{l+1}, \ldots, \mathbf{p}_n$ can be expressed as a combination of the control points $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_l$. So, we deduce that there are $n-l$ redundant control points. In this case, the appropriate pseudo-inverse is:

$$J_{pr}^{+} = (J_{pr}^{T}J_{pr})^{-1}J_{pr}^{T} \tag{4.31}$$

    In this case we have that $(I - J_{pr}^{T}J_{pr}) = 0$, because the dimension of the kernel of J_{pr} is 0. So, the solution can be rewritten as:

$$\widehat{\mathbf{F}_r} = J_{pr}^{+}\mathbf{F}_p \tag{4.32}$$

  - In the second case, if $n < l$ the system is under-constrained, so there are not enough degrees of freedom to uniquely determine the forces that the robots must apply. In this case the appropriate pseudo-inverse is:

$$J_{pr}^{+} = J_{pr}^{T}(J_{pr}J_{pr}^{T})^{-1} \tag{4.33}$$

    In general, for $n < l$ we have $(I - J_{pr}^{T}J_{pr}) \neq 0$, and all of the vector of the form $(I - J_{pr}^{T}J_{pr})\mathbf{w}_l$ belong to the kernel of J_{pr}, so the solution is given by:

$$\widehat{\mathbf{F}_r} = J_{pr}^{+}\mathbf{F}_p + (I - J_{pr}^{+}J_{pr})\mathbf{w}_l \tag{4.34}$$

4.3.1.3 Velocities and accelerations of the robots

After having obtained the forces that the robots must exert on the hose, we have to determine the velocities of the robot contact points which would be a consequence of the desired applied forces, this is an important information because robots control commands are usually given in terms of velocities.

In the preceding section we have obtained an approximation of the forces $\widehat{\mathbf{F}_r}$ that the robots must apply on the contact points with the hose to get the desired motions in the control points, however we will obtain an approximation to the desired forces on the control points $\widehat{\mathbf{F}_p}$:

$$\widehat{\mathbf{F}_p} = J_{pr} \cdot \widehat{\mathbf{F}_r}. \tag{4.35}$$

Introducing $\widehat{\mathbf{F}_p}$ in the Lagrange equation (4.19) we get:

$$M\widehat{\mathbf{A}} = \widehat{\mathbf{F}_p} + \mathbf{P}. \tag{4.36}$$

Using a *LU* decomposition of $\mathbf{M}$ we obtain the accelerations on the control points $\hat{\mathbf{A}}$, and then we obtain the robots accelerations by using again J_{pr}:

$$\widehat{\mathbf{A}^r} = J_{pr}^{+} \cdot \widehat{\mathbf{A}}, \tag{4.37}$$

where $\widehat{\mathbf{A}^r} = [\widehat{\mathbf{a}_r}]$ are the desired acceleration commands for the robots.

We obtain the desired robots velocities for the next step, $\hat{\mathbf{v}}_r(k+1)$, by solving them from equation 4.23.

Since the robots have physical limitations, some velocities and accelerations may not be achievable by a robot, so we define v_m and a_m as the respective maximum of the norm of the velocity and acceleration that a robot can apply. In order to contemplate this limitation in our approach, after having obtained the desired accelerations of the robots we limit them by the rule defined in algorithm 4.1.

The velocities $\hat{\mathbf{v}}_r$ obtained after have applied algorithm 4.1 are used as the commands for the robots.

4.3.2 Hose transportation control

In this section we deal with two ways to perform the hose transportation that can be assumed as different task specifications. The first follows the formal control definition approach of the previous section and the second is

Algorithm 4.1 Physic velocity and acceleration limitations of the robots

1. $\bar{v}$=maximum of the robot velocity norms $\|\hat{\mathbf{v}}_r(k+1)\|$.
2. $\bar{a}$=maximum of the robot acceleration norms $\|\hat{\mathbf{a}}_r(k+1)\|$.
3. if $\bar{v}>v_m$ then
   (a) for each $\hat{\mathbf{v}}_r$ and $\hat{\mathbf{a}}_r$
      i. $\widehat{\mathbf{v}_r(k+1)} = \widehat{\mathbf{v}}_r \frac{v_m}{\bar{v}}$
      ii. $\widehat{\mathbf{a}}_r = \frac{\widehat{\mathbf{v}_r(k+1)} - \widehat{\mathbf{v}_r(k)}}{\triangle t}$
4. if $\bar{a}>a_m$ then
   (a) for each $\hat{\mathbf{v}}_r$ and $\hat{\mathbf{a}}_r$
      i. $\widehat{\mathbf{a}}_r = \widehat{\mathbf{a}}_r \frac{a_m}{\bar{a}}$
      ii. $\widehat{\mathbf{v}_r(k+1)} = \widehat{\mathbf{v}_r(k)} + \widehat{\mathbf{a}}_r \cdot \triangle t$

a formalization of an heuristic approach that has been simulated and latter brought to real life implementation.

4.3.2.1 Following a sequence of intermediate hose configurations

The transportation task is defined by a sequence of intermediate hose configurations h_k that must be passed through by the hose to reach the final destination. Those intermediate configurations can be set by an external control.

In each transition between configurations, defining the reference velocity magnitude as v_{ref}, we attempt to obtain the mean control point velocity vector whose norm $\left\|\bar{\dot{\mathbf{p}}}\right\|$ is as close to v_{ref} as possible, until they reach the desired control points positions $\mathbf{p}_*$. Following the proportional control law of equation 4.22, we define a function $f(.)$ of the current control point positions $\mathbf{p}$, and the velocity magnitude reference v_{ref} that will serve us to obtain a smooth estimation of the sequence of control point vectors that the hose must follows to reach $\mathbf{p}_*$. Then we rewrite the proportional control law as follows:

$$\dot{\mathbf{p}} = k \cdot (\phi(f(v_{ref}, \mathbf{p})) - \mathbf{p}). \tag{4.38}$$

We interpolate the configurations of the sequence, h_k, by a clamped B-interpolating curve, denoted by $\phi(\xi)$, with $\xi \in [0,1]$, $\phi(0) = \mathbf{p}_0$ and $\phi(1) = \mathbf{p}_*$. The number of control points of ϕ, n_k, is the same as the number of configurations of the sequence h_k. Then, the norm of the mean velocity of the control points may be written as:

$$\left\|\bar{\dot{\mathbf{p}}}_\mathbf{i}\right\| = k \left\|\overline{\phi_i(\xi) - \mathbf{p}}\right\|, \; i \in [1, n_k]. \tag{4.39}$$

So, the problem is reduced to find, in every step, the value t that approximates the velocity reference magnitude v_{ref}.

$$f(v_{ref}, \mathbf{p}) = \min_{\xi} \left(v_{ref} - \left\|\overline{\phi_i(\xi) - \mathbf{p}}\right\|\right) \tag{4.40}$$

The interpolating clamped B-spline curve is explained in appendix A.

4.3.2.2 Follow the leader approach

In this section we explain the heuristic control approach to perform the task of transporting a hose by a set of robots applying a strategy of following the leader. The leader trajectory is assumed to be given, and the aim is to define a control strategy for the follower robots. We define a control heuristic taking into account the shape of the hose segments between robots and the distances between consecutive robots. In this approach every robot, except the leader, controls its own velocity and direction based on the shape segment of the hose between it and the foregoing robot. We assume that if a pair of robots are too close, the hose segment between them will describe a curve. In contrast, if they are sufficiently separated, the hose is stretched and approximated to the straight line between the robots.

The idea is that if a the segment of the hose between a pair of adjacent robots is very stretched then the back robot must increase its velocity in order to avoid tension in the hose, and dragging back the foregoing robot. This will relax the hose that will show some curvature. In contrast, if the hose segment is very curved then the back robot must reduce its velocity in order to allow straightening of the hose segment. Figure 4.7 illustrates this relation between the robot distances and the hose segment shape.

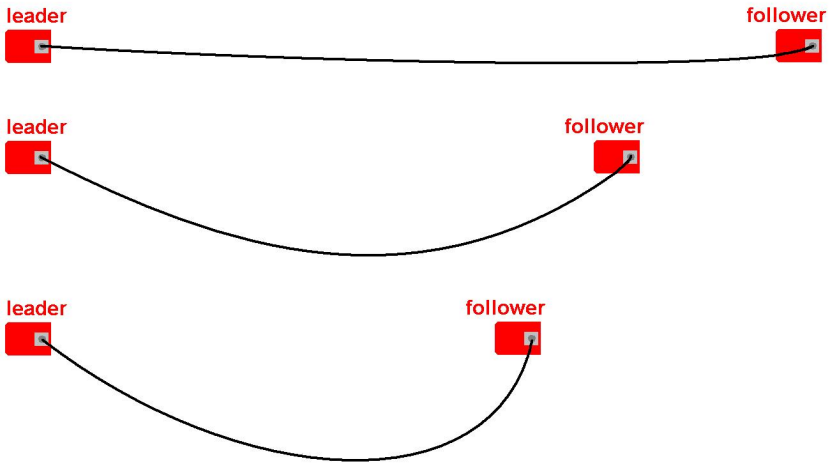


Figure 4.7: Hose segment according to the robots distance.

The curvature of a hose segment is indirectly measured as the proportion between the maximum distance d_h from the hose curve h to the line $L_{\mathbf{r}_1,\mathbf{r}_2}$, defined by the robot's positions ($\mathbf{r}_1$,$\mathbf{r}_2$), and the distance between robots, d_r:

$$\mathfrak{c} = \frac{d_h}{d_r}, \tag{4.41}$$

where

$$d_h = max \left\| h_i - L_{\mathbf{r}_1,\mathbf{r}_2} \right\|, \ \forall h_i \in h,$$

$$d_r = \left\| \mathbf{r}_1 - \mathbf{r}_2 \right\|.$$

The distance from the hose to the straight line defined by the robots positions is computed as the greatest perpendicular distance between the straight line $L_{\mathbf{r}_1,\mathbf{r}_2}$ and the points h_i of the segment hose h, as illustrated in figure 4.8.

Algorithm 4.2 Heuristic control rules for the velocity of a follower robot.

Denote c_i the curvature of segment i and w_{i+1} the velocity magnitude for robot $i+1$.

1. if $\mathfrak{c}_i < \underline{\mathfrak{c}}$ then
   (a) $w_{i+1} = w_2$
2. else
   (a) if $\mathfrak{c}_i > \bar{\mathfrak{c}}$ then
      i. $w_{i+1} = w_0$
   (b) else
      i. $w_{i+1} = w_1$

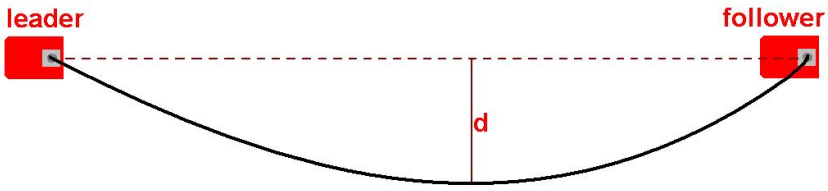


Figure 4.8: Distance of the from the straight line to measure its curvature.

The magnitudes d_h and d_r give us the relation between the dimensions of the rectangle that encloses the hose segment, being d_r the length of the rectangle and d_h its width. We define a maximum and minimum segment curvature for the transport of the hose, denoted by $\underline{\mathfrak{c}}$ and $\bar{\mathfrak{c}}$, and three velocity magnitude levels, w_0, w_1 and w_2 assigning the medium magnitude velocity, w_1, for the leader robot. The follower robots determine their velocities by the heuristic presented in algorithm 4.2.

Besides the control heuristic for the magnitude of the robot velocity, we define a strategy for the velocity direction. We aim to maintain the robots formation in a straight line, so we define in every processing step the velocity direction of the follower robots by an intermediate directions between its current velocity direction and the direction of the previous robot in order to gradually align the robots in the transport of the hose. In figure 4.9 we define

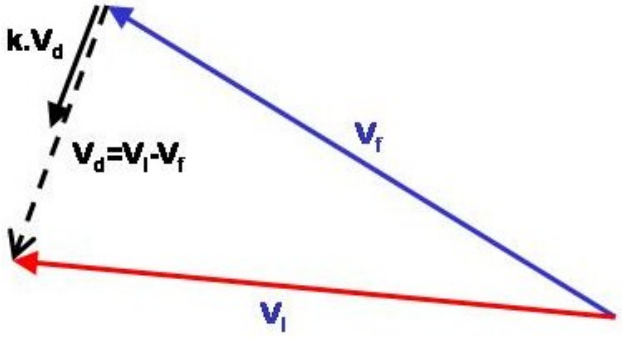


Figure 4.10: Follower robot velocity.

$\mathbf{v}_l$ as the velocity of the leader robot and $\mathbf{v}_f$ as the velocity of the follower robot.

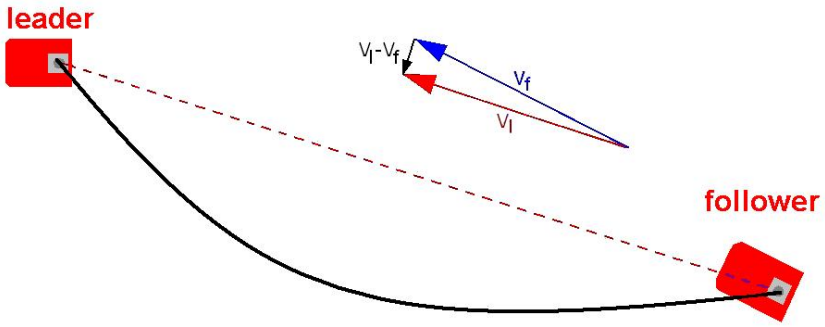


Figure 4.9: Velocity direction for the follower robot.

We modify the velocity direction of a follower robot respect to its previous robot, by the vectorial sum showed in figure 4.10, adding to the follower robot a vector in the direction of the difference between the previous robots direction and its current direction, then we divide this vector by its norm in order to obtain a vector of unit norm.

The new direction for the follower robot velocity is defined by the next equation, where k is a constant that determine the speed at which the follower robot approximate its direction $\mathbf{v}_{i+1}$ to the previous robot direction $\mathbf{v}_i$.

$$\mathbf{v}_{i+1} = \frac{\mathbf{v}_{i+1} + k(\mathbf{v}_i - \mathbf{v}_{i+1})}{\|\mathbf{v}_{i+1} + k(\mathbf{v}_i - \mathbf{v}_{i+1})\|}. \tag{4.42}$$

Time-step	Discretization	Large	Density
1 ms	1 cm	1m	200 g/m

Table 4.1: Hose parameters.

Finally, the velocity of robot i is defined as $w_i \cdot \mathbf{v}_i$.

4.4 MCRS Hose System Simulation

We modeled the hose dynamic in Matlab for 3, 5 and 11 robots. From the dynamic physical model point of view of the hose we only implemented the bending and stretching forces, because we assume that the twisting forces are negligible for the movements of the robots because the grasping of the hose is very tight and the robots will not twist the hose. The parameters of the hose are summarized in table 4.1.

We assumed that robots make decisions of control with a time frequency of $30ms$, so every $30ms$ the robots make decisions about changes in their velocities.

4.4.1 Simulation of hose configuration control

In the simulation of the hose configuration control we start from an initial vector of control points $\mathbf{p_0}$ and the initial robots positions $\mathbf{r_0}$. In every step of the simulation we obtain the velocities of the robots $\mathbf{v_r}$ that decrease the distance from the actual configuration to the the desired hose state as specified by $\mathbf{p}_*$.

From $\mathbf{p_0}$ we obtain an uniform sequence of interpolating points $\mathbf{q_0}$, the vector of knots $\mathbf{U}$ and the vector of robot knots $\mathbf{U_r}$. From $\mathbf{p}_*$ we obtain a sequence of interpolating points, $\mathbf{q}_*$ as inputs of the algorithm 4.3, which contains the steps in the simulation of the control of the hose configuration.

The algorithm 4.3 has some inconvenient:

- The proportional control of the hose may reach a local minimum in which does not exist a decreasing direction that reduce the distance between the current and desired configurations of the hose.
- Some desired configurations of the hose can not be reached because it is impossible to drive the appropriate movements in the robots for

Algorithm 4.3 Simulation of the Hose transportation through configuration control.

1. $h_0(\mathbf{p_0},\mathbf{U},\mathbf{U_r})$= interpolating B-spline $(\mathbf{q_0},\mathbf{U},\mathbf{U_r})$
2. $h_*(\mathbf{p}_*,\mathbf{U},\mathbf{U_r})$=interpolating B-spline$(\mathbf{q}_*,\mathbf{U},\mathbf{U_r})$
3. J_{pr}=compute the Robots Jacobian $(\mathbf{U_r})$
4. repeat while ($\|\mathbf{p}_* - \mathbf{p_0}\|$>tolerance)
   (a) $\mathbf{M}$= MassMatrix
   (b) $\mathbf{P}$= Derivatives of potential energy
   (c) $\mathbf{F}_e$=Generalized external forces
   (d) $\mathbf{V}_{next} = k(\mathbf{p}_* - \mathbf{p})$ <− proportional control law
   (e) $\mathbf{A}$=getAccelerations($V_{current}$,V_{next}) <− physics limitations of the robots
   (f) $\mathbf{F} = \mathbf{MA} - \mathbf{P} - \mathbf{F}_e$
   (g) $\mathbf{F_r} = J_{pr}^{+}\mathbf{F}$
   (h) $\mathbf{A} = \mathbf{M}^{+}[J_{pr}\mathbf{F_r} + \mathbf{P} + \mathbf{F}_e]$
   (i) $\mathbf{V}$=integrateAccelerations($\mathbf{A}$)
   (j) $\mathbf{V}_r = J_{pr}^{+}\mathbf{V}$

obtaining a stable configuration of the hose with minim energy, or there are not enough robots for determining the desired movements in the degrees of freedom of the hose.

Assuming the velocity of every control point of the spline model of the hose can be controlled, a proportional control law might be used in order to obtain the trajectory from the initial configuration to the desired configuration of the hose. The control law is defined on the control points space:

$$\dot{\mathbf{p}} = k(\mathbf{p}_* - \mathbf{p}). \tag{4.43}$$

The trajectories of the robots are approximately a straight line until the robots are near to their final positions, then their trajectories vary in order

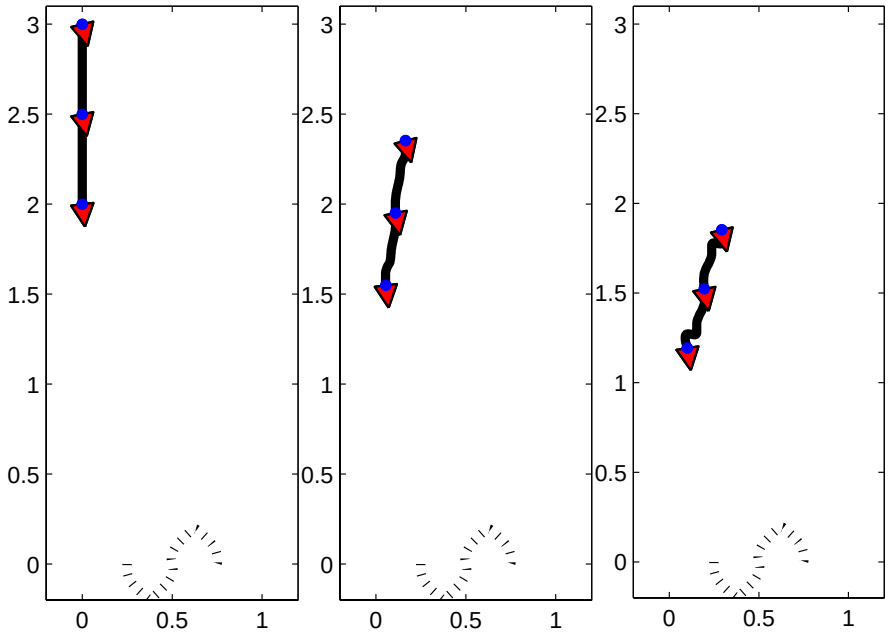
to enhance the contact points positions. Finally the control points converge to their desired positions. A sequence of snapshots of the robot positions and hose configuration along the ideal trajectory is presented in figures 4.11 and 4.12 where we illustrate the transition from the initial configuration of the hose (thick black line) to the desired final configuration (discontinuous black line).

Then, we repeat the experiment but in spite of defining the control law on the control points space we define it on the robot velocities, applying this velocities to the robots in a simulation of a hose including the hose internal dynamics. We define the velocities not taking into account the dynamic of the hose but we apply the obtained velocities for the robots to a hose with internal dynamics. The control law is as follow:

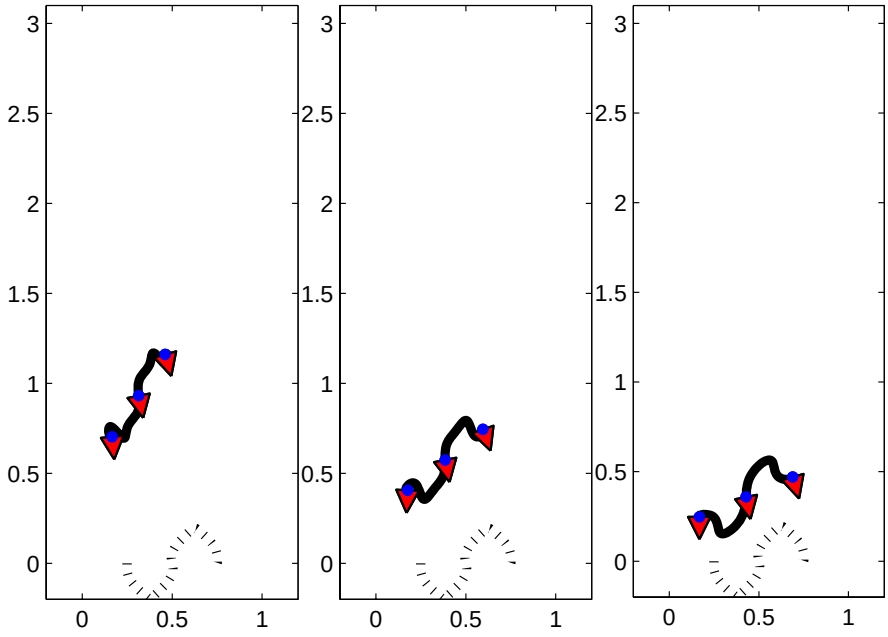
$$\dot{\mathbf{r}} = J_{pr}^{+}[k(\mathbf{p}_* - \mathbf{p})] \tag{4.44}$$

Although we do not take into account the dynamics of the hose when computing the desired control velocities of the robots. We simulate the hose shape evolution as a consequence of its full dynamic model, so the robot's velocities are different in comparison with the previous case because the control points are not placed at the same positions in both approaches. In figure 4.13 the trajectories of robots are presented, where every trajectory is a straight line from the initial position to the final position of the robot. In this case, the robots are not able to fit the control points positions to the desired values, because the matrix J_{pr} is not invertible, so the trajectories of the control points can not be determined by the trajectories of the robots. In figures 4.14 and 4.15 we show the evolution of the hose in this case. It can be appreciated that the robot contact points reach their desired positions, however the shape of the hose does not match the desired shape.

Finally, we use a control law defined in the robots velocities space, but deriving the robots velocities from the dynamic of the hose, as explained in section 4.3.1. The trajectory of the robots for a hose with full model of its internal dynamics, obtaining the robots velocities from the dynamic of the hose, is presented in figure 4.16. Figures 4.17 and 4.18 show the evolution of the hose along the controlled trajectory. Again, the robots reach their desired position while the shape of the hose does not match its desired form. We have built a realistic simulation framework that allows to simulate many of the situations that may happen in the multi-robot-hose interaction. It may be a suitable workbench to evaluate the proposition of robot control algorithms,

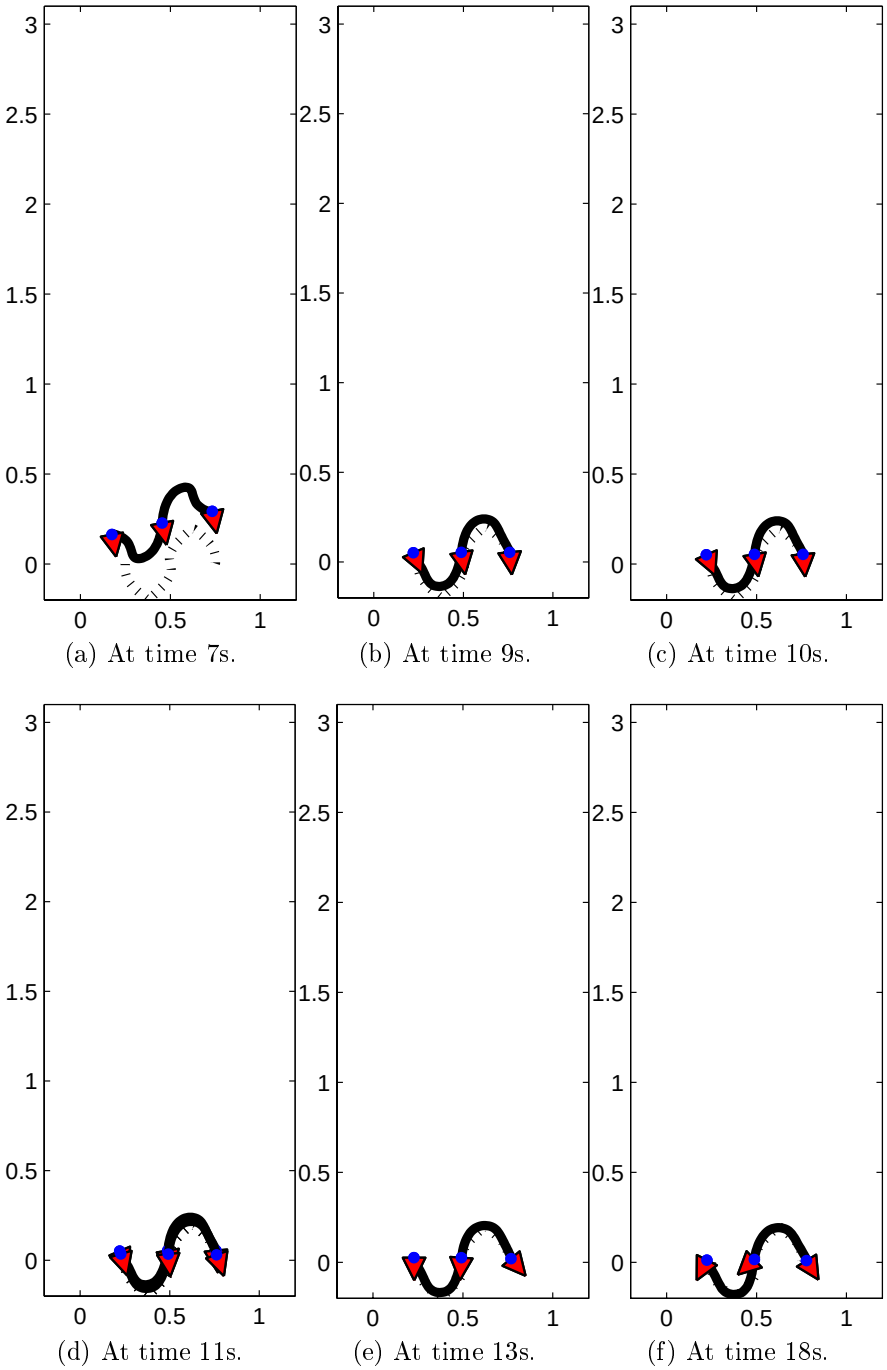


(a) At time 0s. (b) At time 0.5s. (c) At time 1s.



(d) At time 2s. (e) At time 4s. (f) At time 5s.

Figure 4.11: Ideal sequence of the hose without dynamics.



(a) At time 7s. (b) At time 9s. (c) At time 10s.

(d) At time 11s. (e) At time 13s. (f) At time 18s.

Figure 4.12: Ideal sequence of the hose without dynamics (cont.).

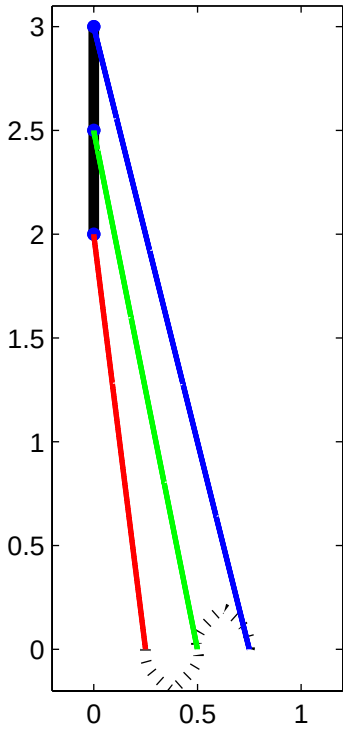
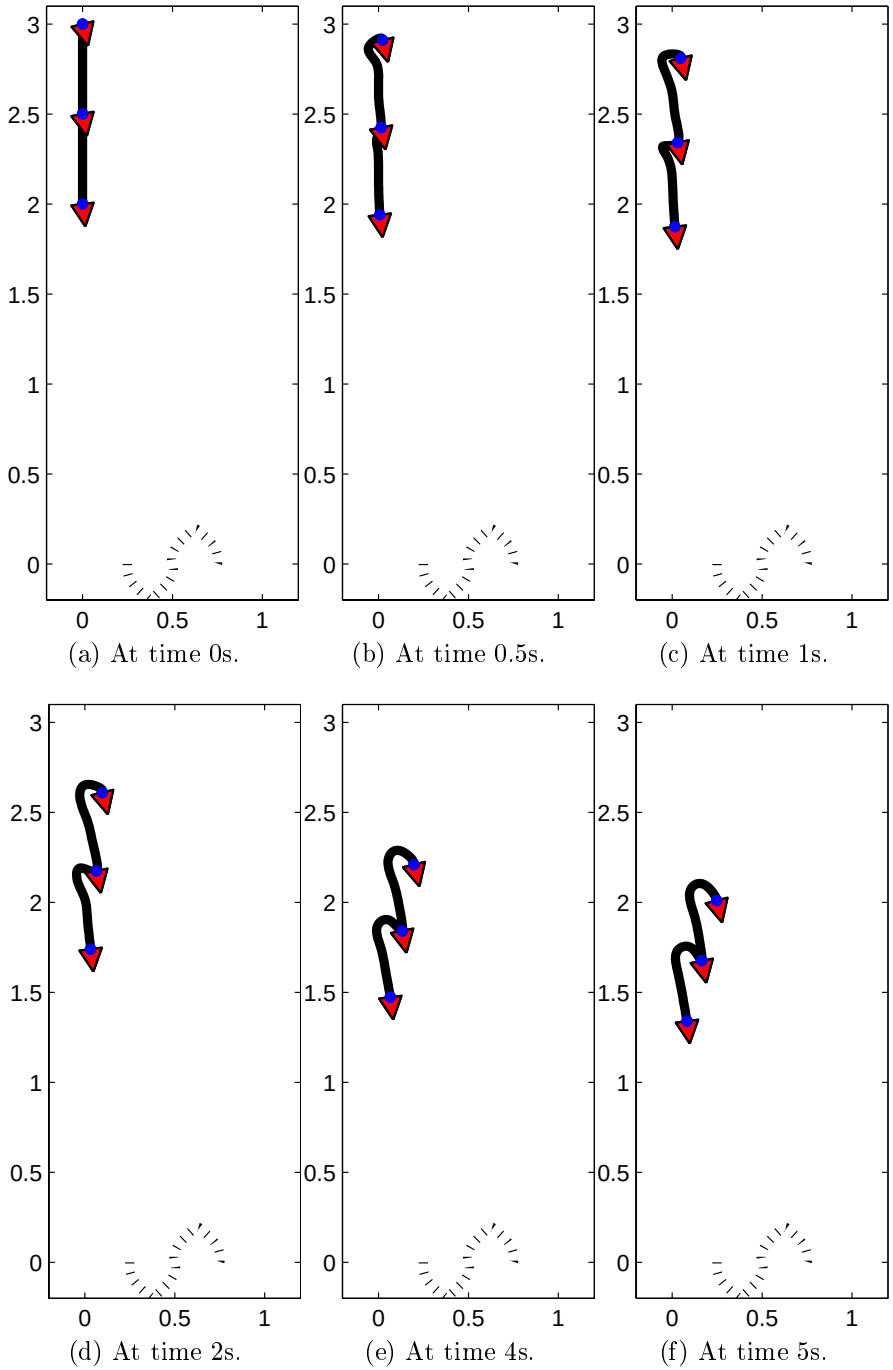


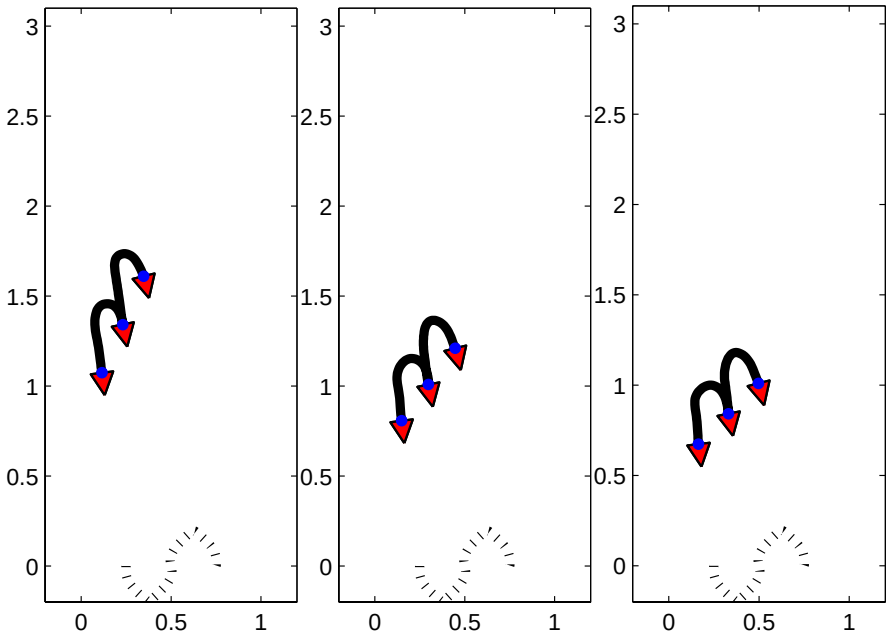
Figure 4.13: Trajectories of the robots without dynamics in the control law.



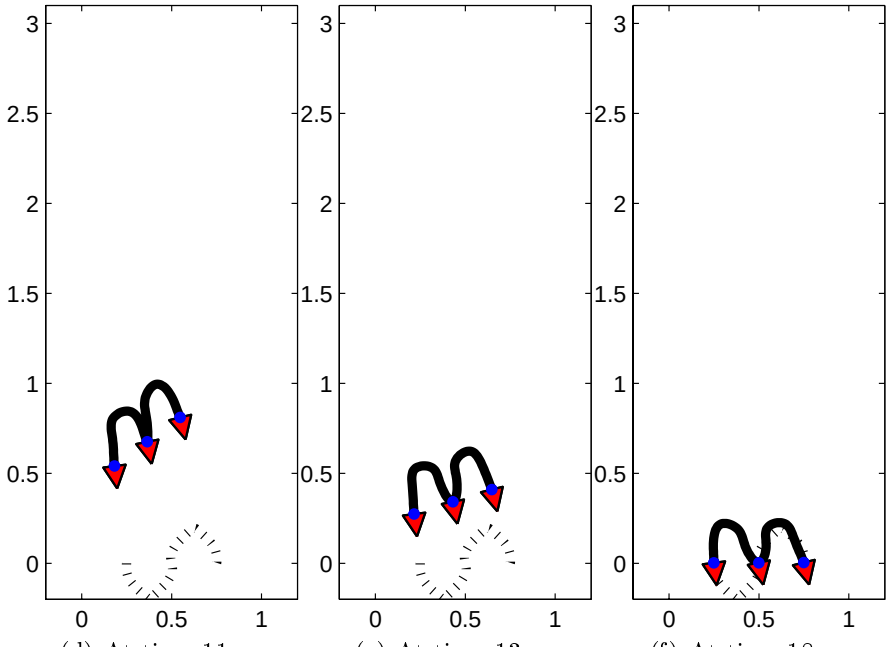
(a) At time 0s. (b) At time 0.5s. (c) At time 1s.

(d) At time 2s. (e) At time 4s. (f) At time 5s.

Figure 4.14: Sequence of the hose with hose internal dynamics.



(a) At time 7s. (b) At time 9s. (c) At time 10s.



(d) At time 11s. (e) At time 13s. (f) At time 18s.

Figure 4.15: Ideal sequence of the hose with hose internal dynamics (cont.).

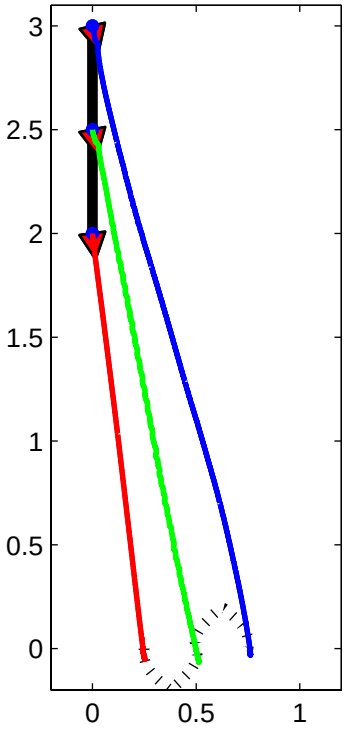


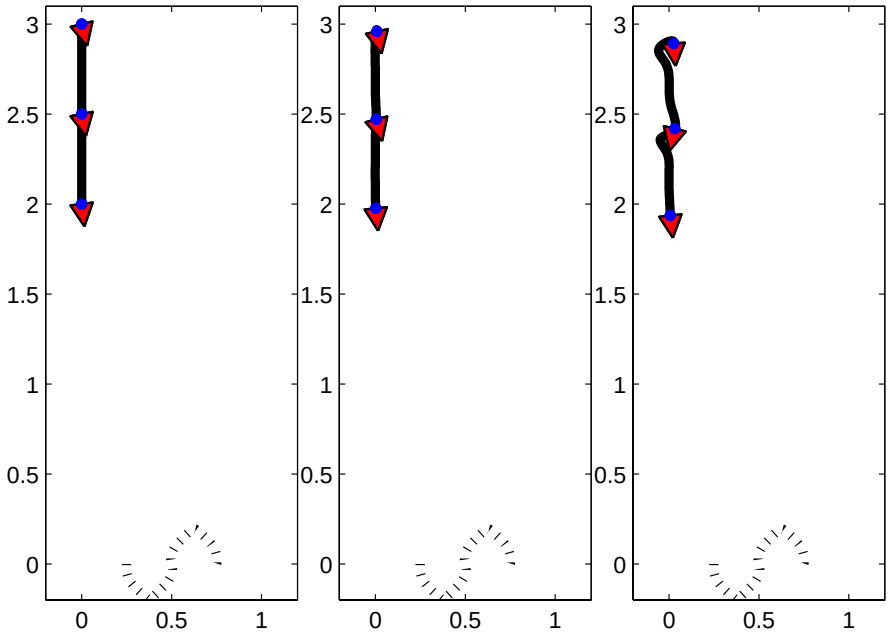
Figure 4.16: Trajectories of the robots from the dynamic in the control law.

as will be shown in the simulation of the heuristic follow-the-leader approach of the next section.

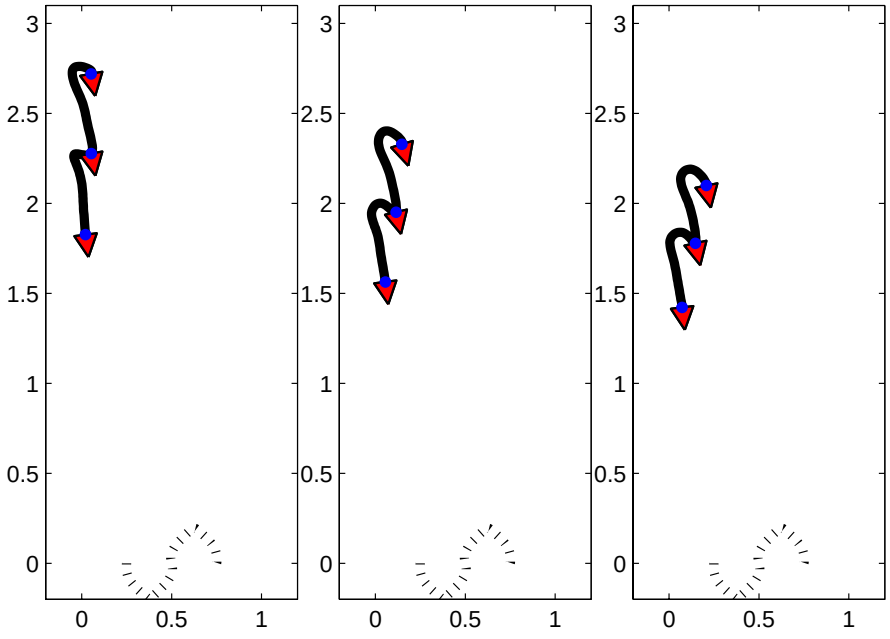
4.4.2 Hose transport control by a heuristic approach

4.4.2.1 Follow the leader approach

First, we implemented the proposed approach for the transport of the hose, this approach has been acquired for a determined range of velocities, but the higher the velocity is, the less the efficiency of this approach. The process for the hose transport control simulation is presented in algorithm 4.4. The main difference between this algorithm and algorithm 4.3 is that the final configuration is not given, the trajectory of the leader is the driving force. At each cycle of the simulation, the velocities of the follower robots are computing according to the leader. In simulation, this trajectory is preset by the user.

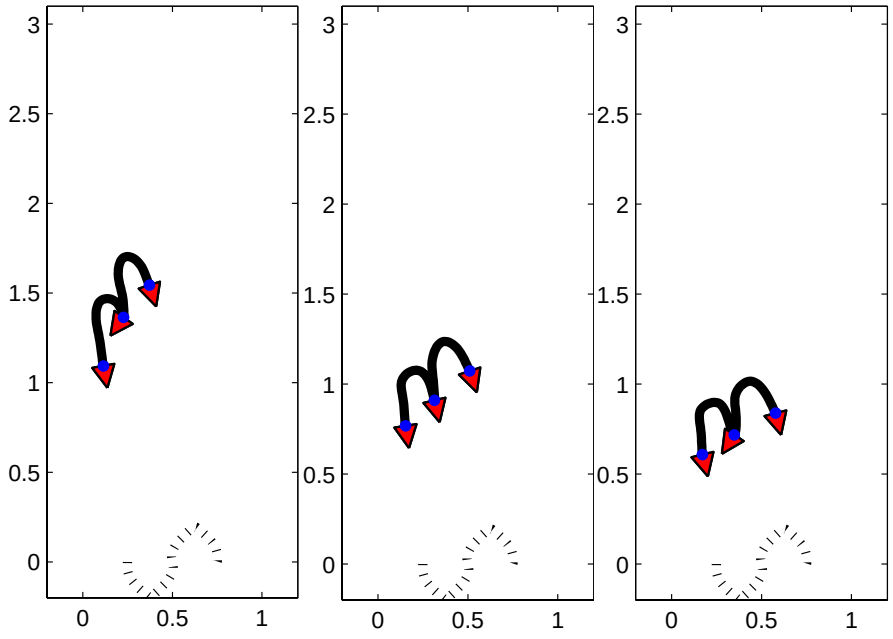


(a) At time 0s. (b) At time 0.5s. (c) At time 1s.

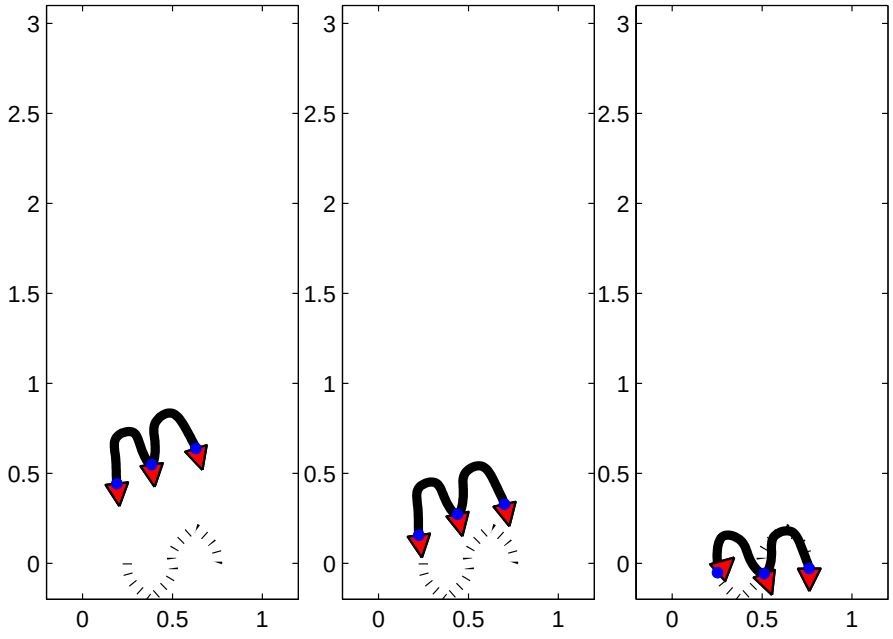


(d) At time 2s. (e) At time 4s. (f) At time 5s.

Figure 4.17: Sequence of the hose with hose internal dynamics and dynamic control law.



(a) At time 7s. (b) At time 9s. (c) At time 10s



(d) At time 11s. (e) At time 13s. (f) At time 18s.

Figure 4.18: Ideal sequence of the hose with hose internal dynamics and dynamic control law(cont.).

Algorithm 4.4 Hose transport control.

1. $h_0(\mathbf{p_0},\mathbf{U},\mathbf{U_r})$= interpolating B-spline $(\mathbf{q_0},\mathbf{U},\mathbf{U_r})$
2. J_{pr}=getRobotsJacobian($\mathbf{U_r}$)
3. loop
   (a) $\mathbf{M}$= MassMatrix
   (b) $\mathbf{P}$= Derivatives of potential energy
   (c) $\mathbf{F}_e$=Generalized external forces
   (d) for each follower robot j
      i. $\mathbf{V}_{r_j}$=getVelocities
      ii. $\mathbf{A}_{r_j}$=getAccelerations($\mathbf{V}_{r_j}$,$\bar{v}$,$\bar{a}$)
   (e) $\mathbf{F} = (\mathbf{M}J_{pr})\mathbf{A}_r - \mathbf{P} - \mathbf{F}_e$
   (f) $\mathbf{F}_r = J_{pr}^{+}\mathbf{F}$
   (g) $\mathbf{A} = \mathbf{M}^{+}[(J_{pr}\mathbf{F_r}) - \mathbf{P} - \mathbf{F}_e]$
   (h) $\mathbf{A}$= fmincon($\mathbf{A}$,J_{pr}^{+},$\mathbf{A}_r$) <– minimization function with constrains
   (i) $\mathbf{p}$=integrateAccelerations($\mathbf{A}$)

One of the interesting phenomenon that we have also observed in the real life, that will be described in section 4.5, is the occurrence of loops in the hose. The bending force, as the force derived from the potential energy that resists to the bending of the hose, only depends on the form of the hose, so its magnitude is the same independently of the velocities that robots apply to the hose. Then, depending on the value of parameter E_b , the maximum allowed distance between robots and, to a lesser degree, on the friction force, the occurrence of loops may be possible, and can be simulated. In figure 4.19 the occurrence of a loop between the leader and the second robot is presented. If the maximum distance allowed is less than the hose segment length, some part of the hose segment will naturally lag behind the follower robot forming a loop. The simulation of this phenomenon and control of the conditions for its occurrence are a good test of the likeliness of our modeling and simulation approach.

Figure 4.19: Hose advancing at robot's velocity of 1m/s.

Increasing the parameter E_b of the bending force, the occurrence of a loop disappeared even maintaining the same velocities and distances between robots as in the previous case, obtaining the constant configuration of the hose when it is stabilized. In figure 4.20 the configuration of the hose is shown, where can be appreciated the form of the hose between the leader robot and its immediately follower without loops. At this value for the bending parameter, the approach considering the hose segments in determining the magnitude of velocities has the desired behavior.

In the following simulation experiments we applied two heuristic approaches to determine the control velocities of the robots for the transport of the hose, by three robots when the leader robot follows a U-shaped trajectory (figure 4.21):

- the approach considering the segment curvature as explained in section 4.3.2.2 and
- a distance-based approach that only takes into account the distance between robots,

In figure 4.21 the robot's trajectories for the distance-based approach are presented, with color red for the leader, color green for the second robot an color blue for the third one.

The trajectory of the leader robot is well tracked by the follower robots, and the adaptive speed depends on the value for the orientation velocity. We also applied the approach considering the distance between robots for 3, 5

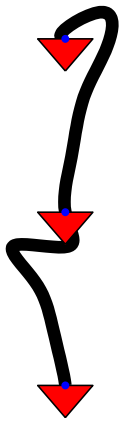


Figure 4.20: Hose advancing at robot's velocity of 0.2m/s.

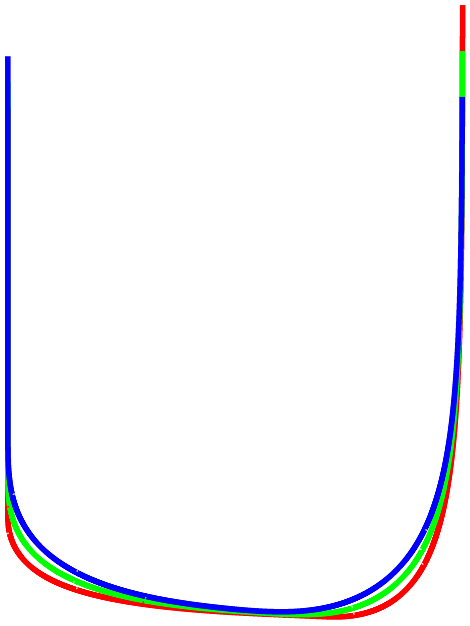
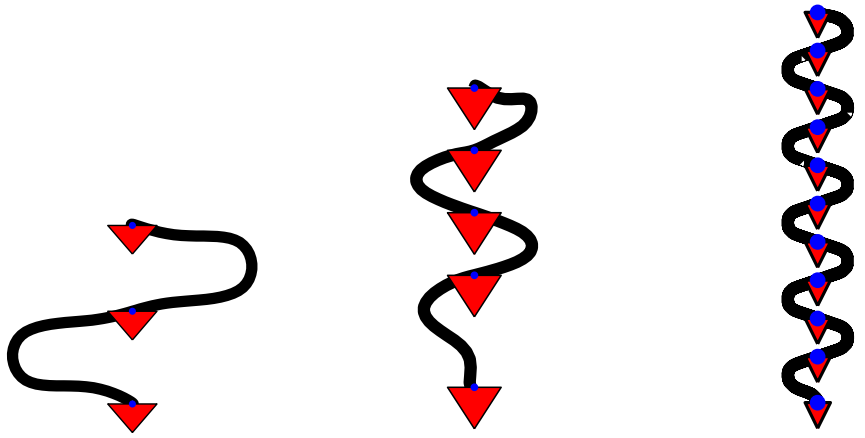


Figure 4.21: Robot's trajectories in the distance based approach.

and 11 robots transporting the hose, starting from the initial configurations shown in figure 4.22.



(a) 3 robots. (b) 5 robots. (c) 11 robots.

Figure 4.22: Hose initial configurations.

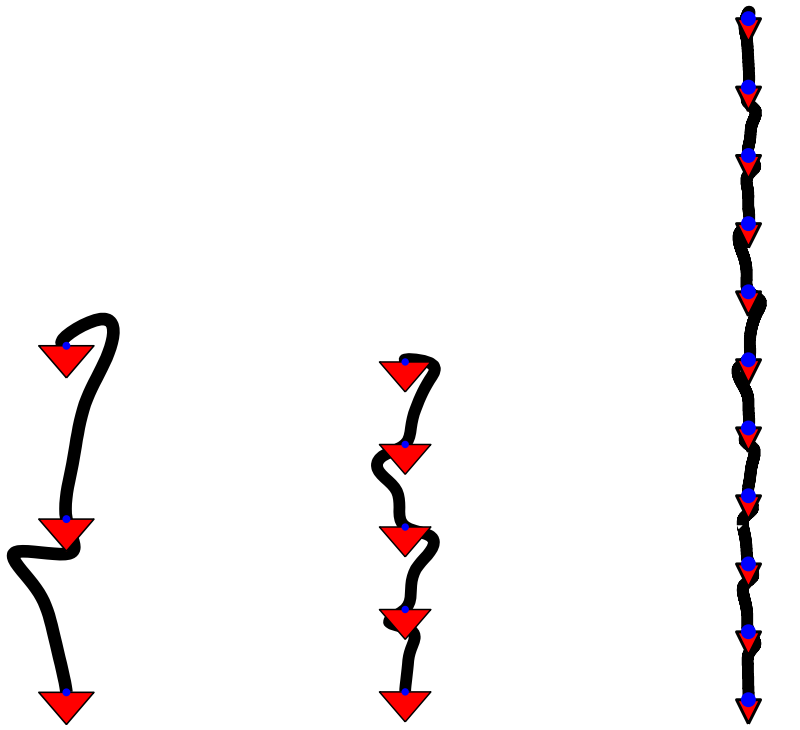
When defining the initial configurations, our aim was to define curved hoses with minimal potential energy in order to get hoses in rest. When all the robots reach the reference velocity, the hose finally stabilizes its configuration. We present sample configurations of the evolution of the hose for the rectilinear advance of the robots with the reference velocity set at 0.2 m/s in figure 4.23 under the distance-based control heuristic.

Some intermediate configurations reached by the hose when transporting it along a U-shaped trajectory using the distance-based heuristic are presented in figure 4.24.

In figure 4.25 we present the velocities of the robots in the x and y axes along the simulation of the transport of the hose following a U-shaped trajectory of the leader robot under the distance-based heuristic.

If we do not limit the accelerations and the forces applied by the robots, the simulation does not have a good behavior, because we force the hoses out the limits that the simulation allows. The forces applied by the robots are presented in figure 4.26.

After we have applied the distance based heuristic approach for the transport of the hose, we apply the heuristic based on the curvature of the hose



(a) 3 robots. (b) 5 robots. (c) 11 robots.

Figure 4.23: Hose rectilinear advance.

segments. The trajectory of the robots for a U-shaped leader trajectory is presented in figure 4.27.

The time plot of the velocities issued as command controls to the robots are presented in figure 4.28. In the segment curvature based approach the velocities for the leader robot are the same as for the distance based approach.

Observing the velocities of the follower robots , we note that the computed velocities are similar for both approaches, but for some local accelerations which are applied until a stable form of the hose segment is reached.

In figure 4.29 the trajectories in the distance based approach and segment curvature based approach are presented together. We notice few differences among them.

4.4.2.2 Configuration trajectory generation

In this section we will consider the simulation of the control of the hose following a sequence of configurations of the hose $\{h_k\}$. We aim to obtain the transport velocities of robots $\mathbf{v}_r$ when following a predefined trajectory of the hose.

As we saw in section 4.3.2.1, the velocities of the control points are determined by the following proportional control law:

$$\left\|\dot{\bar{\mathbf{p}}}_\mathbf{i}\right\| = k \left\|\overline{\phi_i(\xi) - \mathbf{p}}\right\| \tag{4.45}$$

In order to avoid the manual design of the intermediate configurations to simulate this approach we select a sequence of the hose configurations, $\{h_k\}$, from the ones traversed by the hose in the trajectory followed while applying the distance based approach, as described in section 4.4.2.1. We selected 100 configurations of the hose uniformly distributed along the trajectory of the hose, and a speed reference, v_{ref}, of $0.2m/s$. We have selected the same speed as for distance based in order to compare the results of both approaches. The maximum force the robot may apply is defined as $4N$.

In figure 4.30 we present the trajectory of the robots, where according to the criteria used until now, the leader robot is presented in red, the 2nd robot in green and the third robot in blue. In this approach the trajectories of 2nd and 3rd robots are much more closed to the leader robot, this is due to the fact that not only the positions of the robots are considered but also the whole hose curve, then the trajectory is perceived as a whole and not as independent velocities of robots or taking into account the indirect connection done by an external object.

As in the other simulations, we compute control decisions at a $30ms$ frequency, although the time-step of the simulation is 1ms, in order to emulate a control process that takes images from a video camera at a frequency of $30ms$ which would left $30ms$ to the task of computing the control commands for the robots. The time plot of the velocities issued to the robots are presented in figure 4.31. In this simulation the maximum accelerations or velocities where not defined, because the velocities and accelerations of the robots are obtained from the interaction of forces in the hose-robots system. Nevertheless the maximum velocity and acceleration are obeyed by the system through the use of maximum forces for robots.

The forces applied by robots are presented in figure 4.32, where only the 2nd robot reaches the maximum force.

4.5 Real Life Experimentation

We have built a real life experiment showing the transport of a hose by three robots following the hose segment curvature based approach presented in section 4.3.2.2. The whole system implements a primitive version of Visual Servoing, that could be categorized as Position Based Visual Servoing. The Visual Servoing loop is feed from the images taken by a camera placed in a nearly zenital position that observes the system. The experiment has been developed in collaboration with Ivan Villaverde, whose PhD dissertation [79] is complementary to this one in the description of the experiment definition, conditions and results. For instance, the image segmentation has been described in [79]. The control process is centralized in a master computer, and the communication between the master and the slaves (robots) is done via radio-modems over the same channel. The heuristic control implemented in this system is the one described in section 4.3.2.2. It consists on determining the state of the hose from the relation between the distance in the image between robots and the width of the bounding rectangle of the segmented hose piece.

The transport of a hose of 2 meters in length was performed by three robots following the hose segment curvature based approach presented in section 4.3.2.2. The hose curvature limits are defined as $\underline{c} = 0.15$ and $\bar{c} = 0.30$. The velocity levels are defined as $w_1 = 0cm/s$, $w_2 = 1cm/s$ and $w_3 = 2cm/s$.

The image processing is done from the images taken by a camera placed in a nearly zenital position that observes the system, extracting the robot positions and the diverse parts of the hose. The basic trajectory experimented was a straight line. The points of contact between robots and the hose must be fixed but allowing the hose to freely rotate over the robots. The leader robot is attached to the starting end of the hose, the second robot at exactly the middle of the hose length and the third robot at the rear end. The initial configuration of the system may be any configuration meeting the following conditions: (1) the robots are roughly aligned in the direction of advance, with small separations, (2) the robots have similar orientations, (3) the hose can be bended but not as much as to disturb the movements of the robots due to the forces that the hose potential energy configuration exert to the robots. The area in which the robots will move must be obstacle free.

The leader direction and orientation must be followed by the rest of robots. Robots must avoid the formation of loops in the hose, trying to

maintain an appropriate separation from each other. Robots must avoid the excessive stretching of the hose that produces dragging between robots.

The control of the system is computed from a mixture of visual information and readings of each robot in-board compass. The compass information is used to try to match the leader's orientation. The visual information is used to control the distance between the robots as they advance. It has to try to keep them in a straight line formation.The computation of the control commands is done in a centralized computer, but each robot is modeled as an autonomous agent. The orientation of the leader robot will be manually set, but the speed is autonomously controlled. Second and third robot will be autonomously controlled. The visual processing is done in the same central computer.

A configuration of the hose-robots system is presented in figure 4.33, with one robot at each end of the hose and the remaining one attached at the mid-length of the hose. Hardware details of the system can be found in the PhD dissertation of Ivan Villaverde [79], so we will skip them in this document.

4.5.1 Results

An instance of the behavior obtained from the experimentation with the SR1 robots is presented in figure 4.34. The hose is highlighted in red color to show the segmentation results, and each robot is tagged with a label containing its state. The state of a robot may be advancing, stretching or shrinking. The leader robot has the advancing state at every moment, while the followers only have it when the hose segment between them and their preceding robot is not too much stretching not too much shrinking, according to the values defined above ($\underline{c} = 0.15$, $\bar{c} = 0.30$). When the robot is in stretching state the hose segment between it and its previous robot is not enough stretched ($c < \underline{c}$) and the robot reduce its velocity respect to the precedent robot in order to allow a stretching of the hose segment. When the robot is in shrinking state the hose segment between it and its previous robot is so much stretched ($c > \bar{c}$) and the robot increase its velocity respect to the precedent robot in order to allow an increasing of the curvature of the hose segment.

The sequence of images in figure 4.34 shows the following situations in the transport sequence:

- Figure 4.34a: The starting stage of the experiment with the leader robot advancing at cruise speed ($v_1 = w_1$) at the front of the hose

while 2nd and 3th robots are waiting ($v_2 = w_0$, $v_3 = w_0$) until their hose segments are stretched enough ($c_1 = 1$, $c_2 = 0.74$).

- Figure 4.34b: After the leader robot has advancing for a while, the first hose segment is under the maximum limit $\bar{c}$ ($c_1 = 0.27$) so the 2nd robot starts advancing at cruise speed ($v_2 = w_1$). Third robot is still waiting ($v_3 = w_0$) the stretching of its hose segment ($c_2 = 0.67$).

- Figure 4.34c: First hose segment is too much stretched ($c_1 = 0.11$), so the 2nd robot reduces its speed ($v_2 = w_0$) in order to allow the bending of the segment. Third robot is still waiting the stretching of the hose segment ($v_2 = w_0$, $c_2 = 0.6$).

- Figure 4.34d: First segment has curved enough ($c_1 = 0.24$) so the second robot start advancing at cruise speed ($v_2 = w_1$). Second segment has stretched enough ($c_2 = 0.28$), so the third robot start advancing at cruise speed ($v_3 = w_1$).

- Figure 4.34e: First segment curvature ($c_1 = 0.20$) continues between the cruising range, so the second robot maintain the cruise speed ($v_2 = w_1$). Second segment has increased its curvature ($c_2 = 0.32$) more than the maximum limit $\bar{c}$, so the third robot reduces its speed ($v_3 = w_0$) in order to allow a stretching of the hose.

- Figure 4.34f: First segment still maintain its curvature ($c_1 = 0.27$) between the cruising range, so the second robot maintain the cruise speed ($v_2 = w_1$). Second segment has stretched too much ($c_2 = 0.15$), so the third robot increases its speed ($v_3 = w_2$) in order to allow a stretching of the segment.

4.6 Conclusions and future work

In this chapter we have built a geometrical and dynamical model of a unidimensional object and its manipulation by a collection of robots. Such a system is a Linked Multi-component Robotic System, or Linked MCRS. We have formally derived the control rule to compute the velocities at the robot positions in order to move the hose configuration from a given initial position to a desired final position. We have also developed a simulation system for

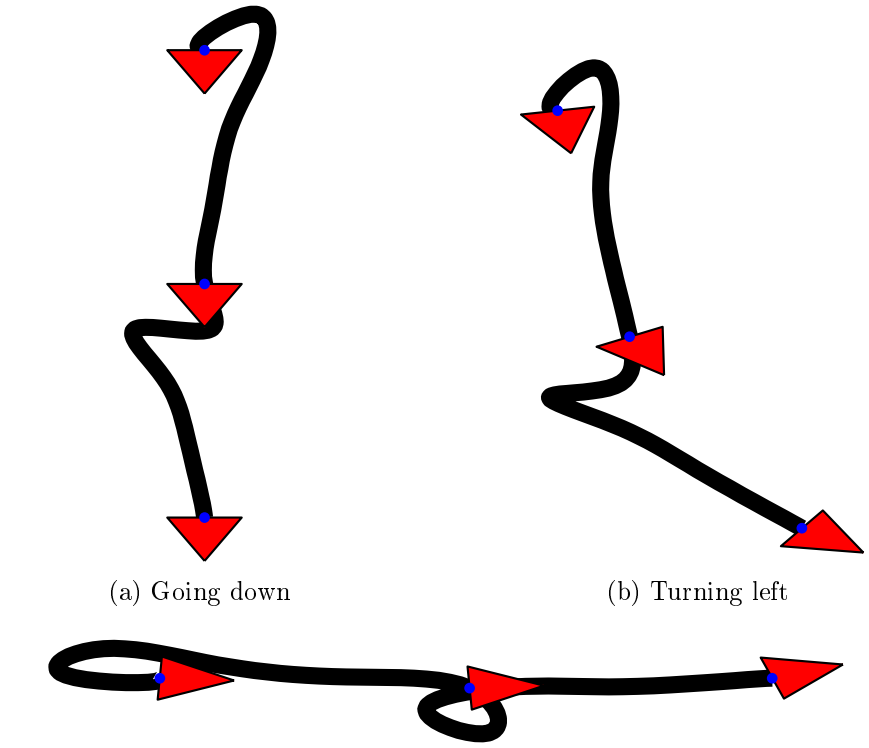
this kind of models, so we can test both the formally derived control algorithms and some heuristic control algorithms in a realistic framework. In this framework we have been able to reproduce some physical phenomena, such as the hose loops, and to test conditions that avoid them. We have, finally, build a physical realization of the system, using a centralized Visual Servoing approach to detect the robots and the hose and to decide the robots control commands. As future work avenues we propose the test and physical realization of distributed control approaches on the MCRS hose system. Algorithms and physical realizations of distributed sensing will enhance the ability of the robots to wander for bigger and less structured spaces, which is the long term goal. Another interesting research area is that of the automated identification of the hose parameters, as a needed calibration step that will prove useful in many instances.

In the following web link, inside of the Computational Intelligent Group web page, we present a sample video of the simulations:

http://www.ehu.es/ccwintco/index.php/DPI2006-15346-C03-03-Resultados-videos-simulacion-manguera

Also, the video from the experiment performed on physicial robots can be found at:

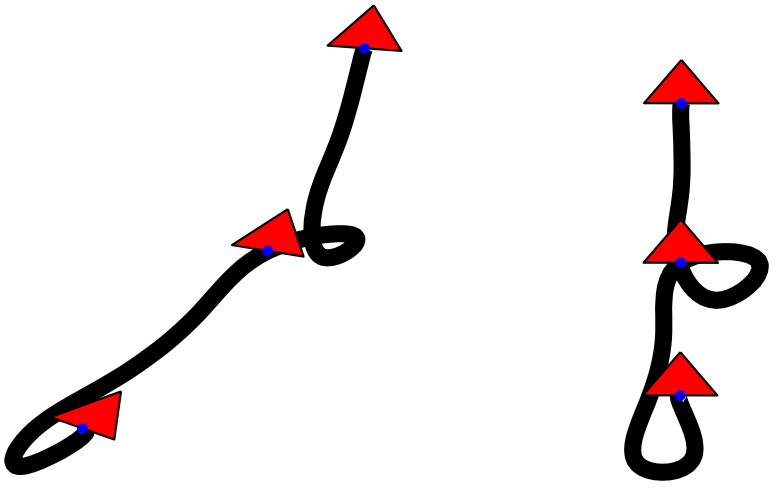
http://www.ehu.es/ccwintco/index.php/DPI2006-15346-C03-03-Resultados-videos-control-centralizado



(a) Going down

(b) Turning left

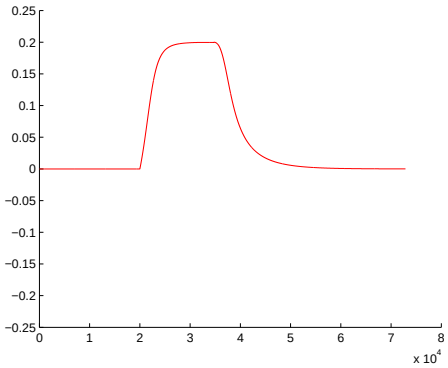
(c) Going right



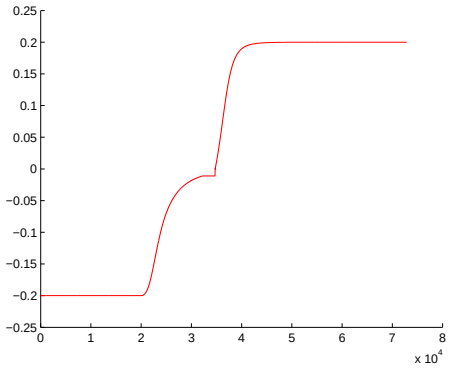
(d) Turning left

(e) Going up

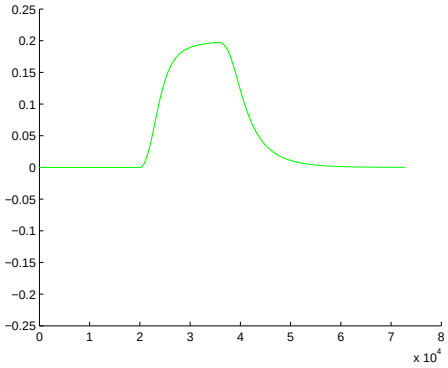
Figure 4.24: Hose configurations along a U-shaped trajectory for distance based heuristic.



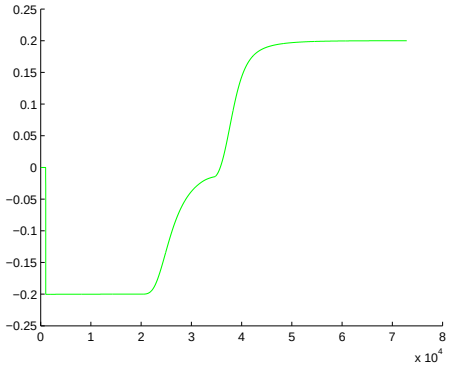
(a) x-axis for leader robot.



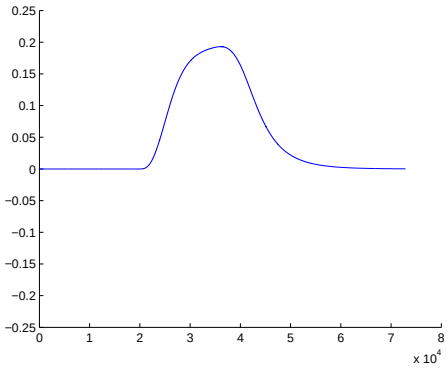
(b) y-axis for leader robot.



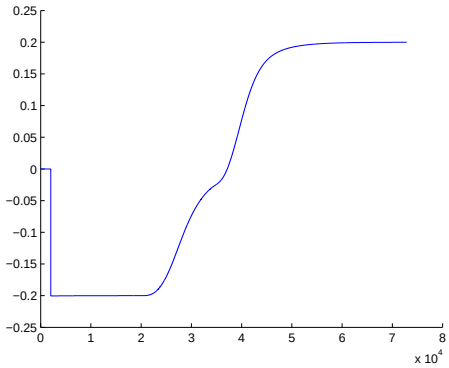
(c) x-axis for second robot.



(d) y-axis for second robot.

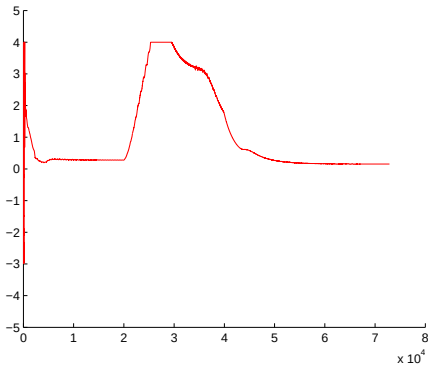


(e) x-axis for third robot.

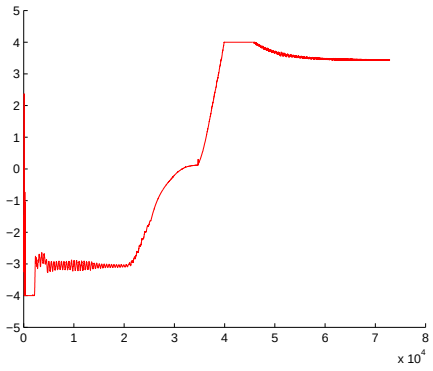


(f) y-axis for third robot.

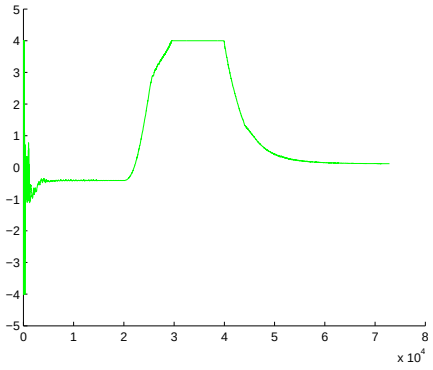
Figure 4.25: Robots velocities during the U-shaped hose transportation for the distance based heuristic approach.



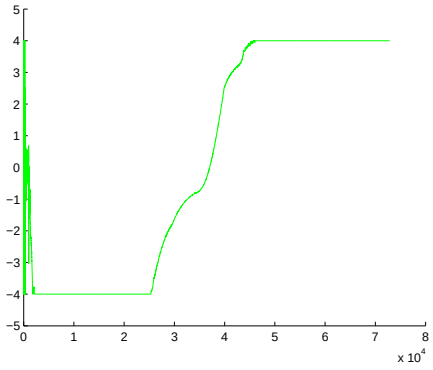
(a) x-axis for leader robot.



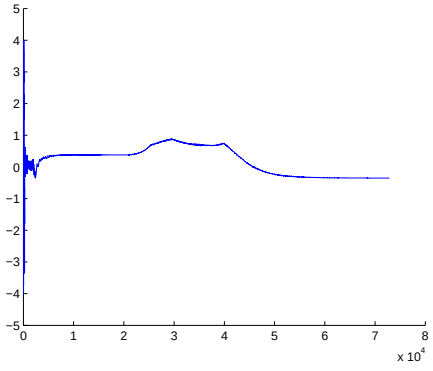
(b) y-axis for leader robot.



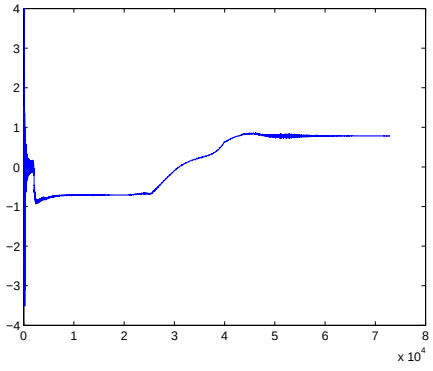
(c) x-axis for second robot.



(d) y-axis for second robot.



(e) x-axis for third robot.



(f) y-axis for third robot.

Figure 4.26: Forces applied by robots for the U-shaped hose transportation under distance based approach.

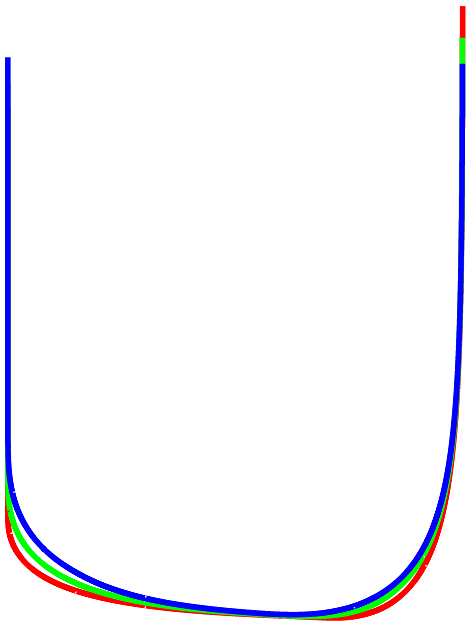
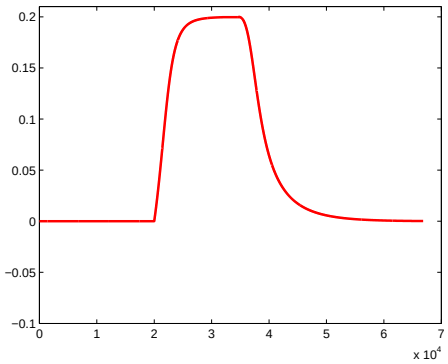
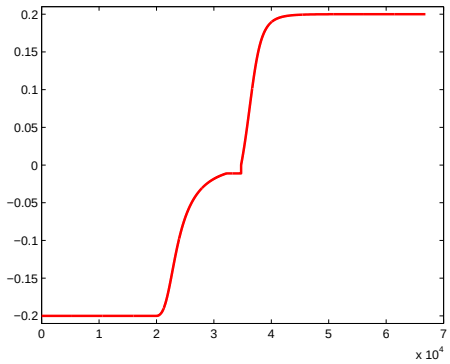


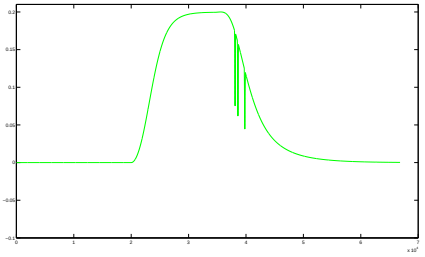
Figure 4.27: Robots trajectories in the segment curvature based heuristic robot control approach.



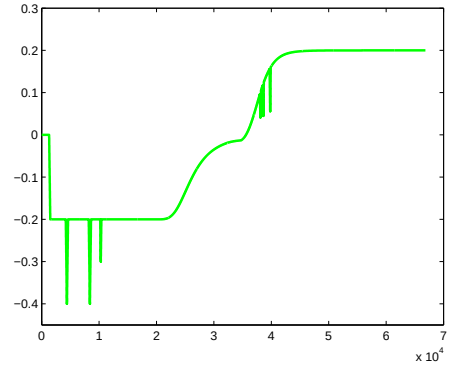
(a) x-axis for the leader robot.



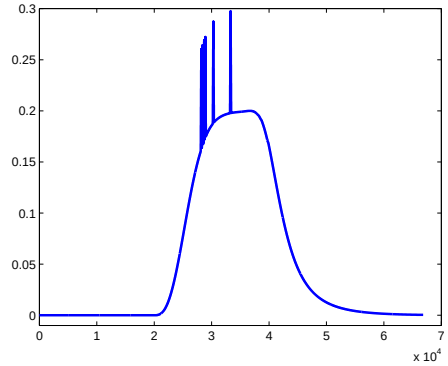
(b) y-axis for the leader robot.



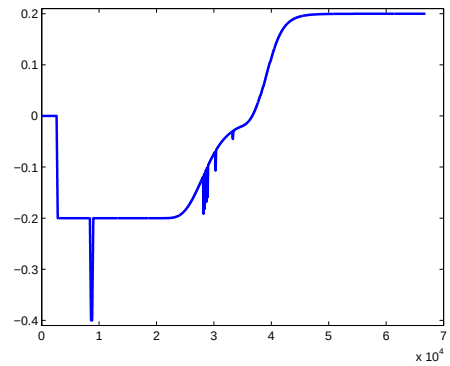
(c) x-axis for second robot.



(d) y-axis for second robot.

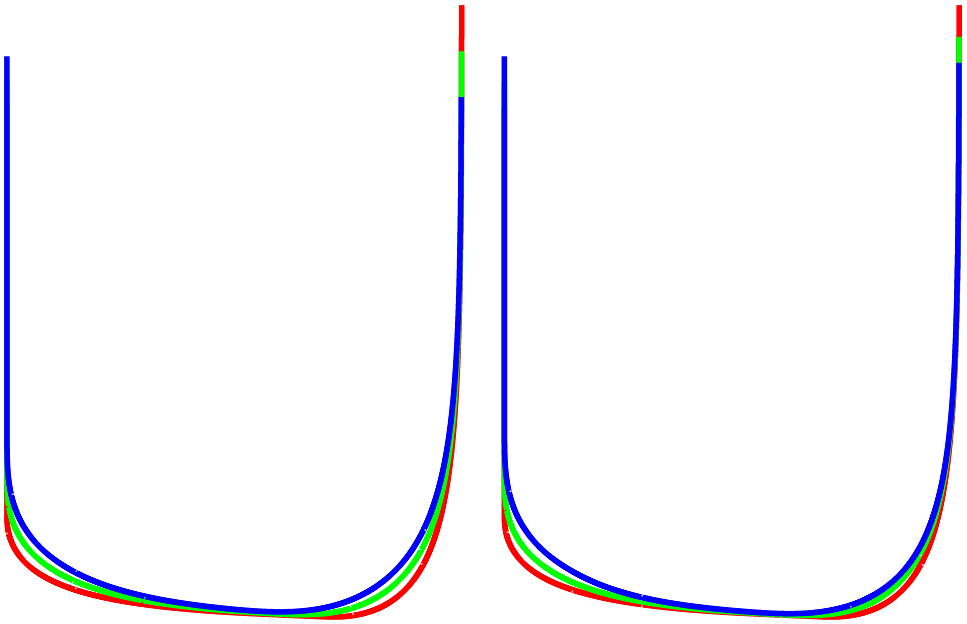


(e) x-axis for third robot.



(f) y-axis for third robot.

Figure 4.28: Robots velocities for the segment based approach.



(a) Distance based. (b) Segment curvature based.

Figure 4.29: Robot's trajectories in the distance and segment curvature based approaches.

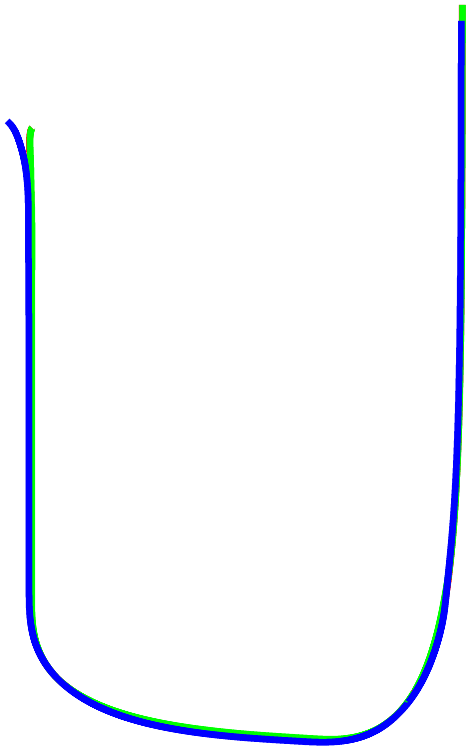
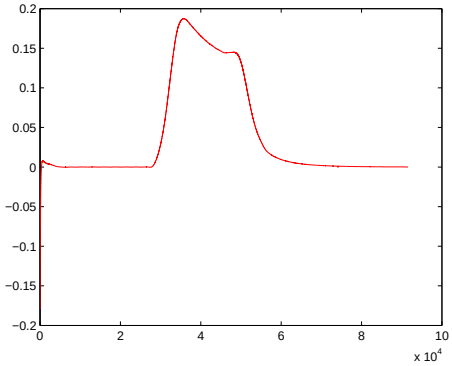
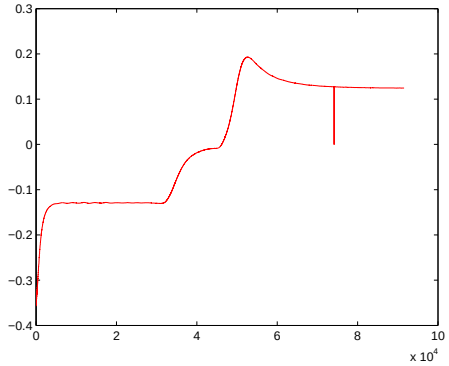


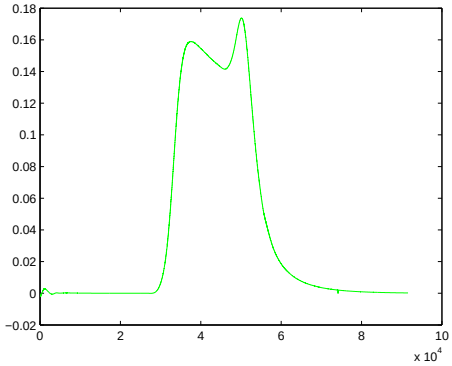
Figure 4.30: Robots trajectory in the configuration trajectory approach.



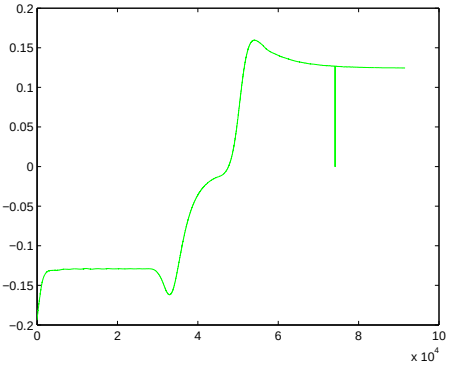
(a) x-velocity for the leader robot.



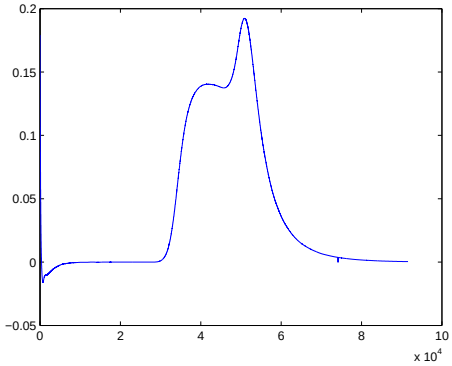
(b) y-velocity for the ladder robot.



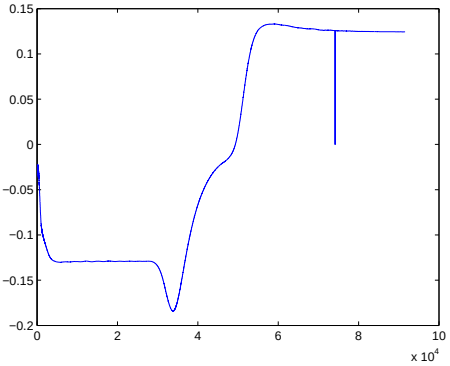
(c) x-velocity for the second robot.



(d) y-velocity for the second robot.

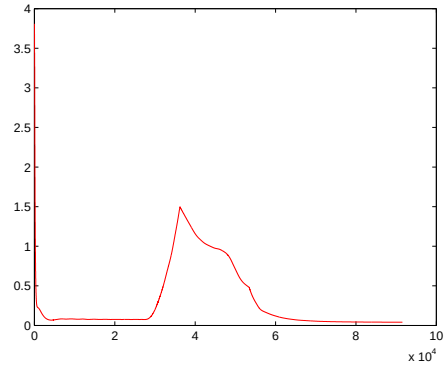


(e) x-velocity for the third robot.

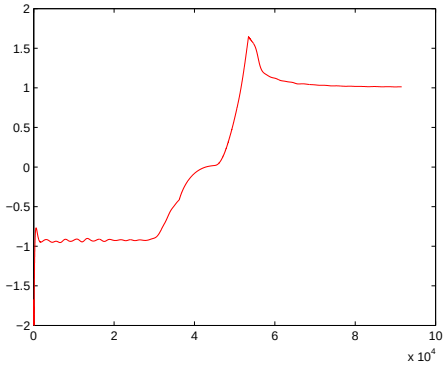


(f) y-velocity for the third robot.

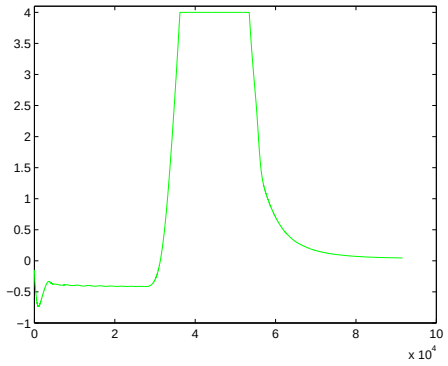
Figure 4.31: Robots velocities in the configuration trajectory approach.



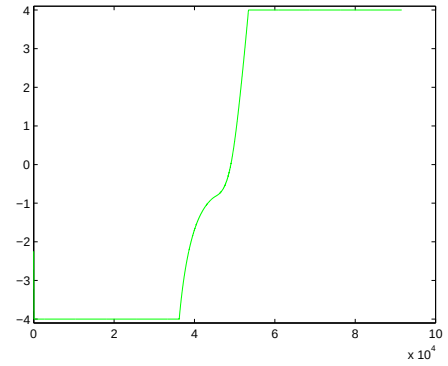
(a) x-axis for leader robot.



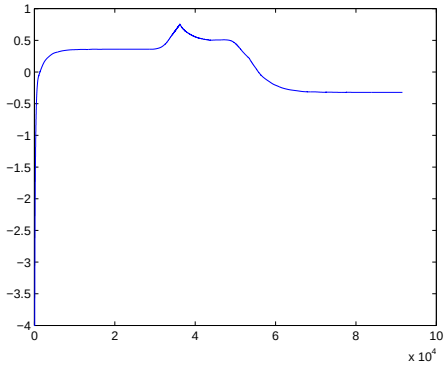
(b) y-axis for leader robot.



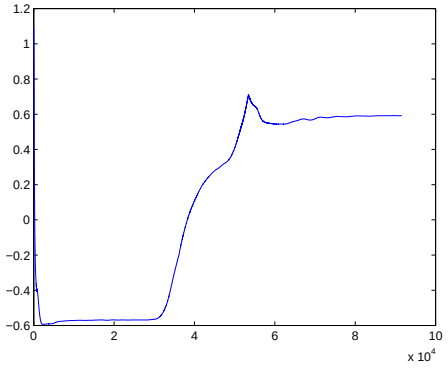
(c) x-axis for second robot.



(d) y-axis for second robot.



(e) x-axis for third robot.



(f) y-axis for third robot.

Figure 4.32: Forces applied by robots for the configuration based approach.

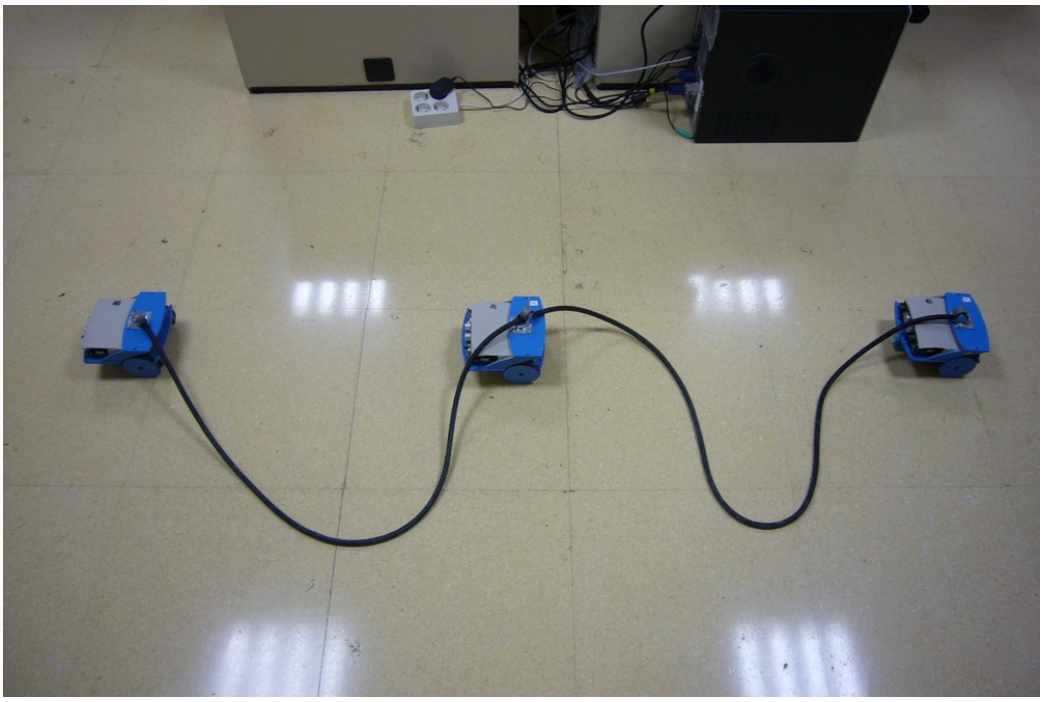
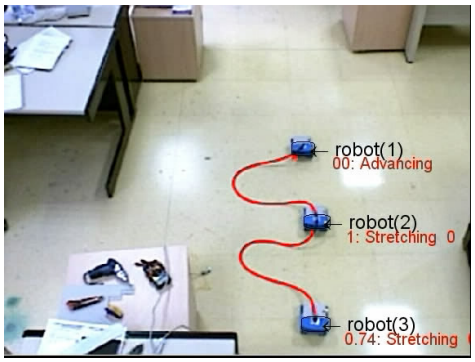
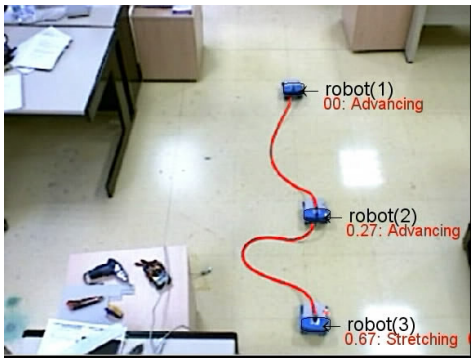


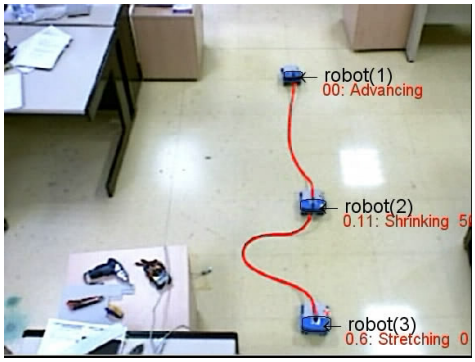
Figure 4.33: Hose-robots physical system.



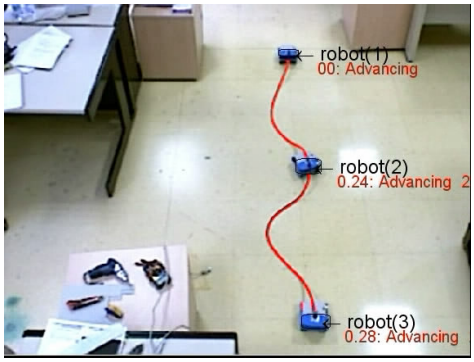
(a) Starting position.



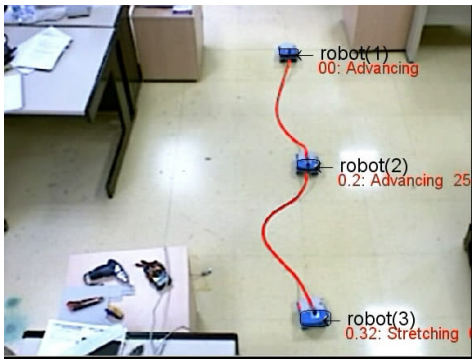
(b) Robot 2 advancing and robot 3 shrinking.



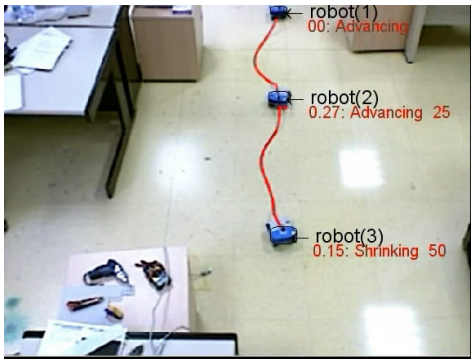
(c) Robot 2 stretching and robot 3 stretching.



(d) Robots 2 advancing and robot 3 advancing.



(e) Robot 2 shrinking and robot 3 advancing.



(f) Robots 2 stretching and robot 3 advancing.

Figure 4.34: Snapshots of the experimentation.

Appendix A

Interpolating clamped B-spline

The interpolating process consists on the construction of a curve that passes through a sequence of preset points in 2D or 3D. Given a set of points $D = \{d_0, ..., d_l\}$, known as interpolating points, there exist infinite curves that pass through these points. In our case we use a clamped B-spline cubic curve and we define the interpolating algorithm for this kind of curves.

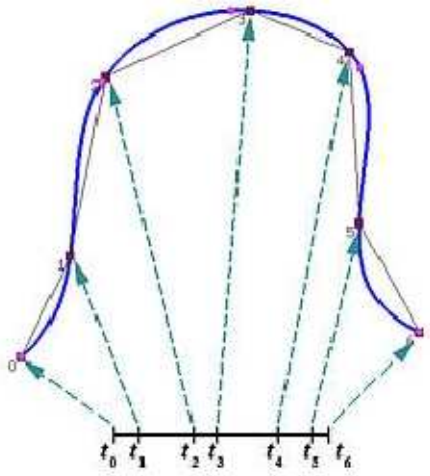


Figure A.1: Interpolating cubic B-spline curve.

We must take into account that the B-spline curves keep the following condition:

$$m = n + p + 1,$$

being $(n+1)$ the number of control points, $(m+1)$ the number of knots and p the curve degree.

A clamped cubic B-spline curve ($p = 3$) has a knots vector:

$$U = \left(\underbrace{u_0, \ldots, u_3}_{4}, \underbrace{u_4, \ldots, u_n}_{n-3}, \underbrace{u_{n+1}, \ldots, u_{n+4}}_{4} \right),$$

where, if the domain of the curve is the interval $[0, 1]$, the knots vector of the clamped B-spline curve is:

$$U = \left(\underbrace{0, 0, 0, 0}_{4}, \underbrace{u_4, \ldots, u_n}_{n-3}, \underbrace{1, 1, 1, 1}_{4} \right).$$

The curve is exclusively defined from $q(u_3)$ to $q(u_{n+1})$ and, for these values of the control parameter u, the curve interpolates the first and last control points. In other words,

$$\begin{aligned} q(u_3) &= p_0 \\ q(u_{n+1}) &= p_n \end{aligned}$$

The first step of the interpolating algorithm consists of selecting the curve set of knots. The most simple knots vector is a non periodic and uniform. If the clamped knots vector is uniform, the values of its knots are computed in the following way:

$$\begin{cases} u_0 = u_1 = u_2 = u_3 = 0 & \\ u_i = \frac{i-3}{n-2} & 4 \leq i \leq n \\ u_{n+1} = u_{n+2} = u_{n+3} = u_{n+4} = 1 & \end{cases},$$

where the difference between two consecutive knots is always constant; in other words:

$$u_{i+1} - u_i = \frac{1}{n-2}.$$

If the interpolating points are not uniformly distributed, these knots vector may get not desired results as peaks, protuberances and loops. This is as consequence of the oscillations of the B-spline, due to the fact that the uniform knots vector does not take into account the geometry of the interpolating points.

Nevertheless, it does not matter which the knots vector is, the curve should interpolate the interpolating points. In other words, the following restrictions must be accomplished:

$$\begin{aligned} d_0 &= q\left(u_3\right) \\ d_1 &= q\left(u_4\right) \\ &\cdots \\ d_{n-2} &= q\left(u_{n+1}\right) \end{aligned}$$

where the k-th point is:

$$d_k = q\left(u_{k+3}\right) = \sum_{i=0}^{n} p_i \cdot N_{i,3}\left(u_{k+3}\right) \qquad 0 \leq k \leq n-2 = l,$$

being

$$u_0, u_1, u_2, u_3 = 0 \qquad u_{n+1}, u_{n+2}, u_{n+3}, u_{n+4} = 1 \ .$$

The relation between l and n is given by:

$$l + 1 = n - 1,$$

where $(l+1)$ is the number of interpolating knots and $(n+1)$ is the number of control points.

When computing these points, $(n+1)$ equations are needed. However, we only have $(l + 1 = n - 1)$ equations. So, two more restrictions are given by repeating the first and last control points of the curve:

$$p_0 = p_1 \text{ y } p_{n-1} = p_n.$$

Given the necessarie restrictions, the equations system is resolved in an effective and efficient way, as we continue explaining. It is remarkable that in the interval $[u_{k+3}, u_{k+4})$, do not exist more than four non-zero basis functions:

$$N_{k,3}\left(u\right) \quad N_{k+1,3}\left(u\right) \quad N_{k+2,3}\left(u\right) \quad N_{k+3,3}\left(u\right) \ .$$

If we examine the expansion in the recursion tree of the Cox-de-Boor algorithm, we can compute efficiently the values of these functions.

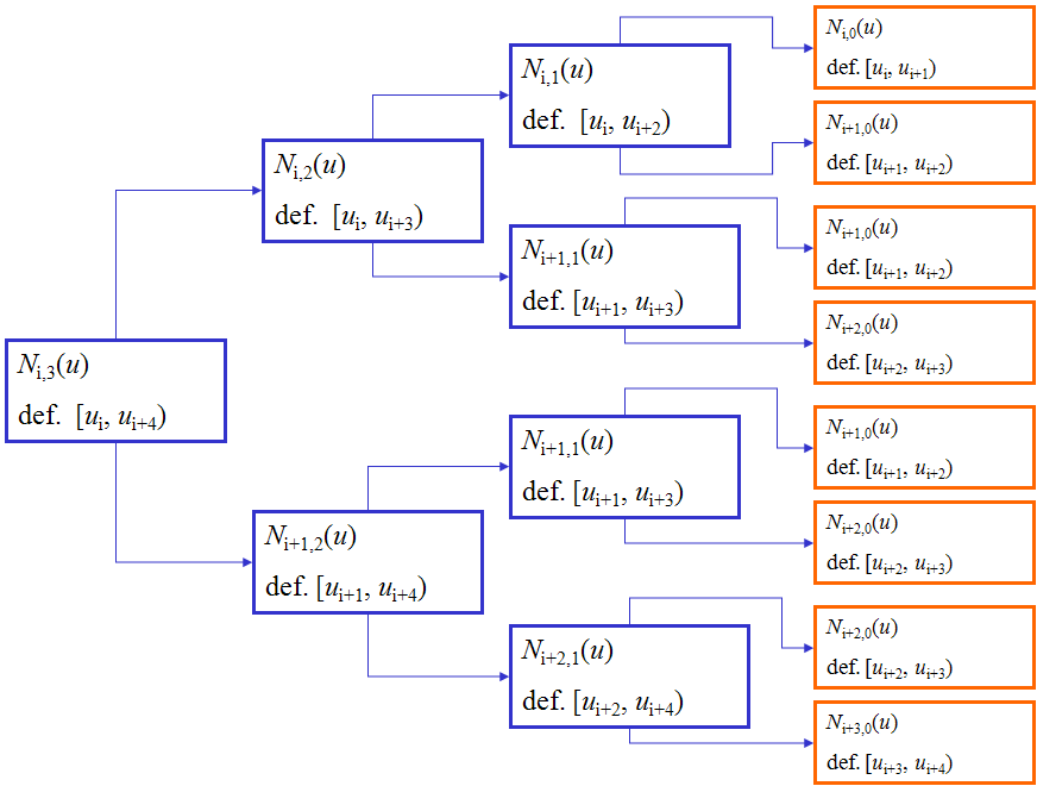


Figure A.2: Recursion tree of the Cox-de-Boor algorithm.

For instance, the three first basis functions take non-zero values while the last basis function is zero. The values of these basis functions are given by:

$$N_{k,3}\left(u_{k+3}\right) = \frac{\left(u_{k+4} - u_{k+3}\right)^2}{\left(u_{k+4} - u_{k+1}\right)\left(u_{k+4} - u_{k+2}\right)} = \alpha_k$$

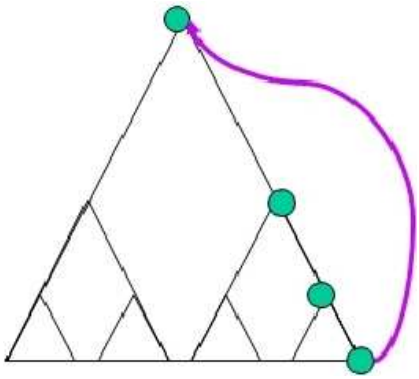


Figure A.3: The recursion tree to calculate the $N_{k,3}$ basis function.

$$N_{k+1,3}\left(u_{k+3}\right) = \frac{\left(u_{k+3} - u_{k+1}\right)\left(u_{k+4} - u_{k+3}\right)}{\left(u_{k+4} - u_{k+1}\right)\left(u_{k+4} - u_{k+2}\right)} + \\ + \frac{\left(u_{k+5} - u_{k+3}\right)\left(u_{k+3} - u_{k+2}\right)}{\left(u_{k+5} - u_{k+2}\right)\left(u_{k+4} - u_{k+2}\right)} = \beta_k$$

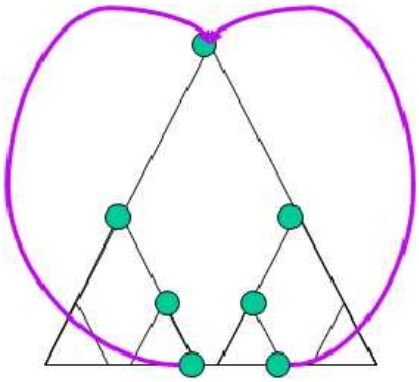


Figure A.4: The recursion tree to calculate the $N_{k+1,3}$ basis function.

$$N_{k+2,3ro}\left(u_{k+3}\right) = \frac{\left(u_{k+3} - u_{k+2}\right)^2}{\left(u_{k+4} - u_{k+2}\right)\left(u_{k+5} - u_{k+2}\right)} = \gamma_k$$

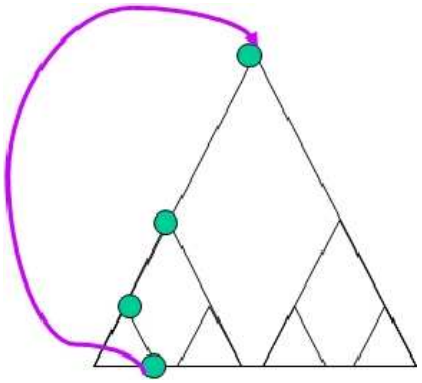


Figure A.5: The recursion tree to calculate the $N_{k+2,3}$ basis function.

So, the interpolating points are given by the following polynomial:

$$\begin{aligned} d_k = q\left(u_{k+3}\right) = N_{k,3}\left(u_{k+3}\right) \cdot p_k + N_{k+1,3}\left(u_{k+3}\right) \cdot \\ \cdot p_{k+1} + N_{k+2,3}\left(u_{k+3}\right) \cdot p_{k+2} = \\ = \alpha_k \cdot p_k + \beta_k \cdot p_{k+1} + \gamma_k \cdot p_{k+2} \end{aligned}$$

and the equations system can be expressed in a matricial way as:

$$\begin{bmatrix} \beta_0 & \gamma_0 & & & & & \\ \alpha_1 & \beta_1 & \gamma_1 & & & & \\ & & \cdots & & & & \\ & & & \alpha_{n-3} & \beta_{n-3} & \gamma_{n-3} & \\ & & & & \alpha_{n-2} & \beta_{n-2} & \end{bmatrix} \cdot \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_{n-2} \\ p_{n-1} \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-3} \\ d_{n-2} \end{bmatrix},$$

being $\beta_0 = \beta_{n-2} = 1$.

The resolution of this equations system obtains the $(n-1)$ control points $\{p_1, \ldots, p_{n-1}\}$ of the B-spline curve. And the remaining two control points, as we have seen previously, are obtained by repeating the first and last control points. This way, given $(l+1)$ interpolating points, the clamped B-spline curve that passes through every interpolating points in the preset order by the user, have the following$(n+1)$ control points :

$$\{p_0 = p_1, p_1, \ldots, p_{n-1}, p_n = p_{n-1}\}.$$

Added to this equations system, that is formulated as matrices for the tri-diagonal systems where only the elements of the main diagonal and its two

neighbors at left and right are non-zero, we can find an efficient algorithm of lineal order ($O(n)$) [80].

Given the following tri-diagonal system of equations:

$$\begin{bmatrix} \beta_0 & \gamma_0 & & & & & \\ \alpha_1 & \beta_1 & \gamma_1 & & & & \\ & & \cdots & & & & \\ & & & \alpha_{n-3} & \beta_{n-3} & \gamma_{n-3} & \\ & & & & \alpha_{n-2} & \beta_{n-2} & \end{bmatrix} \cdot \begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_{n-3} \\ X_{n-2} \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_{n-3} \\ d_{n-2} \end{bmatrix},$$

we can use a two stepped algorithm that resolves this system.

The first step, known as forward step, transforms this tri-diagonal system in the following equations system:

$$\begin{bmatrix} 1 & \lambda_0 & & & & \\ & 1 & \lambda_1 & & & \\ & & & \cdots & & \\ & & & & 1 & \lambda_{n-3} \\ & & & & & 1 \end{bmatrix} \cdot \begin{bmatrix} X_0 \\ X_1 \\ \vdots \\ X_{n-3} \\ X_{n-2} \end{bmatrix} = \begin{bmatrix} \delta_0 \\ \delta_1 \\ \vdots \\ \delta_{n-3} \\ \delta_{n-2} \end{bmatrix}.$$

In order to do so, calculations between the rows of the matrices or between multiples of this rows are done as follows:

$$\lambda_0 = \tfrac{\gamma_0}{\beta_0} \qquad \lambda_i = \tfrac{\gamma_i}{\beta_i - \alpha_i \lambda_{i-1}} \qquad i = 1, \ldots, n-3$$

$$\delta_0 = \tfrac{d_0}{\beta_0} \qquad \delta_i = \tfrac{d_i - \alpha_i \delta_{i-1}}{\beta_i - \alpha_i \lambda_{i-1}} \qquad i = 1, \ldots, n-2$$

The second step, known as backward step, works out the unknown quantities, X_i, by a cycle from backward to forward in the following way:

$$X_{n-2} = \delta_{n-2}$$

$$X_i = \delta_i - \lambda_i \cdot X_{i+1} i = n-3, \ldots, 0$$

Finally, the set of control points is the set of the obtained variables, X_i, from the backward step:

$$p_0 = p_1 \qquad \underset{i=0,\ldots,n-2}{p_{i+1} = X_i} \qquad p_n = p_{n-1} \ .$$

Bibliography

[1] S. S. Antman. *Nonlinear Problems of Elasticity*. Springer-Verlag, 1995.

[2] M. Bonkovic, M. Cecic, and V. Papic. Adaptive neural network (ann) for visual servoing: the mimetic approach. *International Journal of Circuits, Systems and Signal Processing*, 1:259–265, 2007.

[3] Carl De Boor. *A Practical Guide to Splines*. Springer, August 1994.

[4] J. L. Buessler, J. P. Urban, H. Kihl, and J. Gresser. A neural module for the visual servoing of a robotic manipulator. In *IEEE Multiconference on Computational Engineering in Systems Applications (CESA96)*, volume 1, pages 246–251, 1996.

[5] E. Cervera, F. Berry, and P. Martinet. Is 3d useful in stereo visual control? In *IEEE International Conference on Robotics and Automation, ICRA '02, Washington DC, USA*, volume 2, pages 1630–1635, May 2002.

[6] F. Chaumette, E. Malis, and S. Boudet. 2d 1/2 visual servoing with respect to a planar object. In *Workshop on New Trends In Image-Based Robot Servoing, IROS'97, Grenoble, France*, pages 45–52, September 1997.

[7] C. Colombo, E. Kruse, A. M. Sabatini, and P. Dario. Vision-based relative positioning through active fixation and contour tracking. In *Proceedings 2nd International Symp. on Intelligent Robotic Systems, SIRS'94 , Grenoble, France*, pages 319–325, July 1994.

[8] P. I. Corke and S. A. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Transactions on Robotics and Automation*, 17(4):507–515, August 2001.

[9] Peter Corke. *Visual Control of Robot Manipulators - A Review*, volume 7 of *Robotics and Automated Systems*, pages 1–31. World Scientific, 1993.

[10] Sony Corporation. Open-r sdk model information for ers-7, 2003.

[11] Eve Coste-Manière, Philippe Couvignou, and Pradeep K. Khosla. Visual servoing in the task-function framework: A contour following task. *Journal of Intelligent and Robotic Systems*, 12(1):1–21, 1995.

[12] P. Y. Coulon and M. Nougaret. Use of a tv camera system in closed-loop position control of mechanisms. In A. Pugh, editor, *International Trends in Manufacturing Technology ROBOT VISION*, pages 117–127. IFS Publications, 1983.

[13] A. Cretual and F. Chaumette. Positioning a camera parallel to a plane using dynamic visual servoing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS '97, Grenoble, France*, volume 1, pages 43–48, September 1997.

[14] A. Cretual and F. Chaumette. Image-based visual servoing by integration of dynamic measurements. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'98, Louvain, Belgium*, volume 3, pages 1994–2001, May 1998.

[15] Richard J. Duro, Manuel Graña, and Javier de Lope. On the potential contributions of hybrid intelligent approaches to multicomponent robotic system development. *Information Sciences*. Accepted for publication.

[16] Bernard Espiau, François Chaumette, and Patrick Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8:313–326, 1992.

[17] J. T. Feddema and O. R. Mitchell. Vision-guided servoing with feature-based trajectory generation [for robots]. *IEEE Transactions on Robotics and Automation*, 5(5):691–700, October 1989.

[18] N. M. Garcia and E. Malis. Preserving the continuity of visual servoing despite changing image features. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2004*, volume 2, pages 1383–1388, September-October 2004.

[19] N. Garcia-Aracil, E. Malis, R. Aracil-Santonja, and C. Perez-Vidal. Continuous visual servoing despite the changes of visibility in image features. *IEEE Transactions on Robotics*, 21(6):1214–1220, Dec. 2005.

[20] C. Gaskett, L. Fletcher, A. Zelinsky, and E. Zelinsky. Reinforcement learning for visual servoing of a mobile robot. In *Proc. of the Australian Conference on Robotics and Automation (ACRA2000)*, pages 149–154, 2000.

[21] C. Geschke. A robot task using visual tracking. *Robotics Today*, pages 39–43, Winter 1981.

[22] Mireille Grégoire and Elmar Schömer. Interactive simulation of one-dimensional flexible parts. *Computer-Aided Design*, 39(8):694–707, 2007.

[23] Greg Hager. The "xvision" system: A general purpose substrate for real-time vision-based robotics. In *Proceedings of the Workshop on Vision for Robotics*, pages 56–63, 1995.

[24] M. Hao and Z. Sun. Image based visual servoing using takagi-sugeno fuzzy neural network controller. In *IEEE International Symposium on Intelligent Control*, pages 53–58, 2007.

[25] H. Hashimoto, T. Kubota, M. Sato, and F. Harashima. Visual servo control of robotic manipulators based on artificial neural network. In *15th Annual Conference of IEEE Industrial Electronics Society, IECON '89*, volume 4, pages 770–774, November 1989.

[26] E. Hergenröther and P. Dhne. Real-time virtual cables based on kinematic simulation. In *Proceedings of the WSCG*, 2000.

[27] G. Hermann, P. Wira, and J. P. Urban. Neural networks organizations to learn complex robotic functions. In Michel Verleysen, editor, *11th European Symposium on Artificial Neural Networks (ESANN2003)*, pages 33–38, 2003.

[28] J. Hill and W. T. Park. Real-time control of a robot with a mobile camera. In *Proceedings of the 9th ISIR*, pages 233–246, Washington D.C., March 1979.

[29] N. Hogan. Impedance control - an approach to manipulation. i - theory. ii - implementation. iii - applications. *ASME Transactions Journal of Dynamic Systems and Measurement Control B*, 107:1–24, March 1985.

[30] K. Hosoda, K. Igarashi, and M. Asada. Adaptive hybrid visual servoing/force control in unknown environment. In *IEEE Int. Conf. on Intelligent Robots and Systems, Osaka, Japan*, pages 1097–1103, September 1996.

[31] K. Hosoda, M. Kamado, and M. Asada. Vision-based servoing control for legged robots. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3154–3159, 1997.

[32] K. Hosoda, T. Miyashita, S. Takeuchi, and M. Asada. Adaptive visual servoing for legged robots-vision-cued swaying oflegged robots in unknown environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 778–784, 1997.

[33] K. Hosoda, K. Sakamoto, and M. Asada. Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3d reconstruction. In *Proceedings of the International Conference on Intelligent Robots and Systems, IEEE/RSJ, IROS '95*, volume 3, pages 29–34, Washington, DC, USA, October 1995. IEEE Computer Society.

[34] S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.

[35] G. Loreto, Wen Yu, and R. Garrido. Stable visual servoing with neural network compensation. In *Proceedings of the 2001 IEEE International Symposium on Intelligent Control (ISIC '01)*, pages 183–188, 2001.

[36] A. G. Makhlin. Stability and sensitivity of servo vision systems. In *Proc 5th International Conference on Robot Vision and Sensory Controls - RoViSeC 5*, pages 79–89, October 1985.

[37] E. Malis, F. Chaumette, and S. Boudet. Multi-cameras visual servoing. In *IEEE International Conference on Robotics and Automation, ICRA '00, San Francisco, CA, USA*, volume 4, pages 3183–3188, April 2000.

[38] Enzio Malis. Visual servoing invariant to changes in camera intrinsic parameters. *IEEE Transactions on Robotics and Automation*, 1:704–709, 2001.

[39] Enzio Malis, Patrick Rives, Vincent Bandou, Anne-Gaelle Allais, and Michel Perrier. Active stereovision using invariant visual servoing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2006*, pages 2326–2331, 2006.

[40] Ezio Malis. *Stability Analysis of Invariant Visual Servoing and Robustness to Parametric Uncertainties*. Springer Berlin/Heidelberg, 2003.

[41] P. Martinet and E. Cervera. Stacking jacobians properly in stereo visual servoing. In *IEEE International Conference on Robotics and Automation, ICRA'01, Seoul, Korea*, volume 1, pages 717–722, May 2001.

[42] N. Maru, H. Kase, S. Yamada, A. Nishikawa, and F. Miyazaki. Manipulator control by using servoing with the stereo vision. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems '93, IROS '93*, volume 3, pages 1866–1870, July 1993.

[43] Y. Mezouar and F. Chaumette. Path planning in image space for robust visual servoing. In *IEEE International Conference on Robotics and Automation, ICRA '00, San Francisco, CA, USA*, volume 3, pages 2759–2764, April 2000.

[44] Y. Mezouar and F. Chaumette. Path planning for robust image-based control. *IEEE Transactions on Robotics and Automation*, 18(4):534–549, August 2002.

[45] W. T. Miller. Sensor based control of robotic manipulators using a general learning algorithm. *IEEE Transactions on Robotics and Automation*, RA-3(2):157–165, 1987.

[46] W. T. Miller. Real-time application of neural networks for sensored-based control of robots with vision. *IEEE Transactions on Systems, Man and Cybernetics*, 19(4):825–831, 1989.

[47] G. Morel, E. Malis, and S. Boudet. Impedance based combination of visual and force control. In *IEEE International Conference on Robotics*

and Automation, Leuven, Belgium, volume 2, pages 1743–1748, May 1998.

[48] Bradley Nelson, James Morrow, and Pradeep Khosla. Robotic manipulation using high bandwidth force and vision feedback. *Mathematical and Computer Modelling - An International Journal*, 24(5/6):11–29, 1995.

[49] Dinesh K. Pai. Strands: Interactive simulation of thin solids using cosserat models. *Computer Graphics Forum*, 21(3):347–352, 2002.

[50] V. Panwar and N. Sukavanam. Neural network based controller for visual servoing of robotic hand eye system. *Engineering Letters*, 14:167–175, 2007.

[51] J. S. Park and M. J. Chung. Trajectory generation for image-based visual servoing using uncalibrated stereo rig. In *Proceedings of International Conference on Automation, Robotics and Computer Vision, Singapore*, volume 12, 2000.

[52] J. S. Park and M. J. Chung. Image-based trajectory generation for visual-servoing using projective invariants. In *Proceedings of International Symposium on Robotics, Seoul, Korea*, volume 4, pages 972–981, 2001.

[53] Jae Seok Park and Myung Jin Chung. Image space trajectory generation for image-based visual servoing under large pose error. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1159–1164, 2001.

[54] J. Pomares and F. Torres. Movement-flow-based visual servoing and force control fusion for manipulation tasks in unstructured environments. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 35(1):4–15, Feb. 2005.

[55] R. E. Prajoux. Visual tracking. In D. Nitzan et al., editor, *Machine intelligence research applied to industrial automation*, pages 17–37. SRI International, August 1979.

[56] M. Prats, R. Ramos-Garijo, P. J. Sanz, and A. P. Del Pobil. Recent progress in the uji librarian robot. In *IEEE International Conference*

on Systems, Man and Cybernetics, volume 6, pages 5227–5232, October 2004.

[57] Hong Qin and Demetri Terzopoulos. D-nurbs: A physics-based framework for geometric design. Technical Report 1, Los Alamitos, CA, USA, 1996.

[58] P. Questa, E. Grossmann, and G. Sandini. Camera self orientation and docking maneuver using normal flow. In *SPIE AeroSense95, Orlando, Florida, USA*, pages 274–283, April 1995.

[59] M. Quinlan, C. Murch, T. Moore, R. Middleton, L. Li, R. King, and S. Chalup. The 2004 nubots team report. Technical report, 2004.

[60] M. H. Raibert and J. J. Craig. Hybrid position/force control of manipulators. *ASME Journal of Dynamic Systems, Measurement, and Control*, 102(2):126–133, 1981.

[61] Th. Röfer, H.-D. Burkhard, U. Düert, J. Homann, D. Göhring, M. Jüngel, M. Lötzsch, O. v. Stryk, R. Brunn, M. Kallnik, M. Kunz, S. Petters, M. Risler, M. Stelzer, I. Dahm, M. Wachter, K. Engel, A. Osterhues, C. Schumann, and J. Ziegler. Germanteam robocup 2003. Technical report, http://www.robocup.de/germanteam/GT2003.pdf, 2003.

[62] Patrick Rives, François Chaumette, and Bernard Espiau. Visual servoing based on a task function approach. In *Experimental Robotics I, Proceedings of the First International Symposium on Experimental Robotics, Montreal, Canada*, pages 412–428, June 1990.

[63] C. Rosen, D. Nitzan, G. Agin, A. Bavarsky, G. Gleason, J. Hill, D. McGhie, and W. Park. Machine intelligence research applied to industrial automation. Technical Report NSF Grant APR-75-13074, SRI Project 4391, 6th Report, SRI International, Menlo Park, CA, November 1976.

[64] C. Rosen, D. Nitzan, G. Agin, A. Bavarsky, G. Gleason, J. Hill, D. McGhie, and W. Park. Machine intelligence research applied to industrial automation. Technical report, 8th Report, SRI International, August 1978.

[65] M. B. Rubin. *Cosserat Theories: Shells, Rods and Points*. Kluwer, 2000.

[66] J. R. Ryoo, D. Y. Kim, and M. J. Chung. An on-line trajectory generation for control of active head-eye system. In *Proceedings of Asian Control Conference, Shanghai, China*, pages 2983–2988, July 2000.

[67] Claude Samson, Michel Le Borgne, and Bernard Espiau. Robot control: The task function approach. In *Oxford Engineering Science Series*, volume 22. Clarendon Press, Oxford University Press, UK, 1 edition, April 1991.

[68] A. C. Sanderson and L. E. Weiss. Image-based visual servo control using relational graph error signals. In *Proceedings of the IEEE International Conference on Cybernetics and Society*, volume 1, pages 1074–1077, 1980.

[69] Joris De Schutter and Johan Baeten. *Integrated Visual Servoing and Force Control: The Task Frame Approach*, volume 8. Springer Tracts in Advanced Robotics, 2003.

[70] Y. Shirai and H. Inoue. Guiding a robot by visual feedback in assembling tasks. *Pattern Recognition*, 5(2):99–108, June 1973.

[71] N. T. Siebel and Y. Kassahun. Learning neural networks for visual servoing using evolutionary methods. In *Proceedings of the Sixth International Conference on Hybrid Intelligent Systems, HIS '06*, page 6, 2006.

[72] I. H. Suh and T. W. Kim. Fuzzy membership fuction based neural networks with applications to the visual servoing of robot manipulators. *IEEE Transactions on Fuzzy Systems*, 2:203–220, 1994.

[73] I. H. Suh and T. W. Kim. A visual servoing algorithm using fuzzy logics and fuzzy neural networks. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA'96*, volume 3, pages 3605–3612, 1996.

[74] I. H. Suh, T. W. Kim, S. Heu, and S. Oh. Visual servoing by a fuzzy reasoning method. In *Proceedings of the IEEE/RSJ International Workshop on Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, IROS '91*, volume 1, pages 111–116, 1991.

[75] V. Sundareswaran, P. Bouthemy, and F. Chaumette. Exploiting image motion for active vision in a visual servoing framework. *International Journal of Robotics Research*, 15(6):629–645, 1996.

[76] A. Theetten, L. Grisoni, C. Andriot, and B. Barsky. Geometrically exact dynamic splines. *Computer-Aided Design*, 40(1):35–48, January 2008.

[77] J. P. Urban, J. L. Buessler, and J. Gresser. Modular neurocontrollers for reaching movements. In *IEEE International Conference on Systems, Man, and Cybernetics*, volume 2, pages 1714–1719 vol.2, Oct 1998.

[78] J. P. Urban, J. L. Buessler, P. Wira, and J. Gresser. Neuromodule-based visual servoing of a robot arm with a 2 d.o.f. camera. In *IEEE International Conference on Systems, Man, and Cybernetics, 'Computational Cybernetics and Simulation'*, volume 4, pages 3507–3512, 1997.

[79] I. Villaverde. *On Computational Intelligence Tools for Vision Based Navigation of Mobile Robots*. PhD thesis, University of the Basque Country UPV/EHU, 2009.

[80] Alan Watt and Mark Watt. *Advanced animation and rendering techniques*. ACM, New York, NY, USA, 1991.

[81] Guo-Qing Wei and G. Hirzinger. Multisensory visual servoing by a neural network. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 29(2):276–280, Apr 1999.

[82] L. E. Weiss. *Dynamic Visual Servo Control of Robots: An Adaptive Image-Based Approach*. PhD thesis, Carnegie-Mellon University, April 1984.

[83] William M. Wichman. Use of optical feedback in the computer control of an arm. Technical report, Standford AI project, AI memo 55, August 1967.

[84] www.tekkotsu.org.

[85] W. Yu and X. Li. Visual servoing with velocity observer and neural compensation. In *IEEE International Symposium on Intelligent Control*, volume 1, pages 454–459, 2004.

[86] W. Yu and M. A. Moreno-Armendariz. Robust visual servoing of robot manipulators with neuro compensation. *Journal of the Franklin Institute*, 342(7):824–838, 2005.