Application of visual servoing to the dynamic positioning of an underwater vehicle

Jean-François Lots

Application of visual servoing to the dynamic positioning of an underwater vehicle

Jean-François Lots

Thesis submitted
for the
Degree of Doctor of Philosophy

Heriot-Watt University
Department of Computing and Electrical Engineering

Contents

List of Figures

List of Tables

List of Abbreviations

UUV	Unmanned Underwater Vehicle
ROV	Remotely Operated Vehicle
AUV	Autonomous Underwater Vehicle
d.o.f.	degree(s)-of-freedom
LMedS	Least Median of Squares
RANSAC	RANdom SAmpling Consensus
SONAR	SOund NAvigation and Ranging
HWU	Heriot-Watt University
OSL	Ocean Systems Laboratory
MBARI	Monterey Bay Aquarium Research Institute
IFREMER	Institut Français de Recherche et d'Exploitation de la MER
resp.	respectively
vs.	versus

List of Symbols

Chapter 2

$\boldsymbol{\nu} = [\boldsymbol{\nu}_p^T, \boldsymbol{\nu}_o^T]^T$:	vehicle's position and orientation 6-vector,
$\boldsymbol{\eta} = [\boldsymbol{\eta}_p^T, \boldsymbol{\eta}_o^T]^T$:	vehicle's linear and angular velocity 6-vector,
$\mathbf{J}(\boldsymbol{\eta}_o)$:	Jacobian matrix of the vehicle,
x, y, z:	horizontal position of an UUV in the Earth-fixed frame
ϕ, θ, ψ:	attitude of the vehicle in the Earth-fixed frame
u, v, w:	surge, sway and heave linear velocities
p, q, r:	roll rate, pitch rate and yaw rate
$\mathbf{M}$:	mass matrix
$\mathbf{C}$:	centrifugal matrix
$\mathbf{D}$:	ANGUS' laminar flow vector
$\mathbf{g}$:	hydrostatic wrench
$\mathbf{B}$:	drag matrix
$\mathbf{E}$, $\mathbf{F}$:	thrust matrices
$\boldsymbol{\tau}$:	ANGUS' thrust matrices
$\dot{X}_c$, $\dot{Y}_c$, $\dot{Z}_c$:	sea current velocity components in Earth-fixed frame

Chapter 3

X_p, Y_p	horizontal position of the Cartesian robot
K_x, K_y	proportional gains
K_{ix}, K_{iy}	integrators gains

Chapter 4

$\mathbf{M}$:	3-D Euclidean point (3-vector)
$\tilde{\mathbf{M}}$:	3-D projective point (4-vector)
$\mathbf{m}$:	2-D Euclidean point (2-vector)
$\tilde{\mathbf{m}}$:	2-D projective point (3-vector)
${}^1\mathbf{T}_2$:	homogeneous matrix co-ordinated frame transformation from frame 1 to frame 2
${}^1\mathbf{R}_2$:	rotation from frame 1 to frame 2

${}^1\mathbf{t}_2$:	translation from frame 1 to frame 2
$\mathbf{C}$:	pinhole camera or optical centre
$\mathbf{A}$:	camera calibration matrix
f:	camera focal length
α_x, α_y:	focal lengths in pixel
s:	skew
x_o, y_o:	principal point co-ordinates in pixel
$\tilde{\mathbf{P}}$:	camera projection matrix
$\tilde{\mathbf{P}}_o$:	canonical projective matrix
$\mathbf{e}$, $\mathbf{e}'$:	epipoles in first and second view
$\mathbf{F}$:	fundamental matrix
$\mathbf{E}$:	essential matrix
$\mathbf{Q}^+$:	pseudo inverse of matrix $\mathbf{Q}$
$\mathbf{H}$:	homography matrix
$\mathbf{n}^T$:	normal to a plane

Chapter 5

u, v:	pixel co-ordinates of an image point
$I(\mathbf{x}, t)$:	image sequence
$\delta(\cdot)$:	2-D motion field
$I_x = \partial I / \partial x$:	partial derivative of I with respect to x

Chapter 6

$\mathbf{e}$:	task function
$\mathcal{F}$, $\mathcal{F}^*$:	current and desired camera frames
$\tilde{\mathbf{m}}_n$:	normalised 2-D point
$\mathbf{L}$:	interaction matrix
${}^c\mathbf{R}_d$:	rotation matrix from $\mathcal{F}$ to $\mathcal{F}^*$
$\mathbf{l}$:	rotation axis of ${}^c\mathbf{R}_d$
Ω:	rotation angle of ${}^c\mathbf{R}_d$
$\boldsymbol{\nu}_s$:	sensor (camera) velocity vector
$\boldsymbol{\nu}_r$:	controllable velocity vector
$\mathbf{L}_r$:	reduced order interaction matrix

$\mathbf{K}_P, \mathbf{K}_I, \mathbf{K}_D$: PID controller gain matrices

Chapter 7

$\boldsymbol{\nu} = [u, v]^T$: reduced-order body-fixed velocity vector (surge and sway)

$\dot{\boldsymbol{\eta}} = [\dot{x}, \dot{y}]^T$: reduced-order Earth-fixed velocity vector

$\mathbf{M}$: reduced-order mass matrix of ANGUS

$\mathbf{B}$: reduced-order hydrodynamic drag matrix of ANGUS

$\mathbf{D} = [D_u, D_v]^T$: reduced-order laminar flow vector

a_1, a_2, a_3: thrusters' efficiency coefficients

u_c, v_c: sea current velocity in body-fixed reference frame

$\mathbf{K}_p, \mathbf{K}_i, \mathbf{K}_d$: PID controller gain matrices

β: surge thrusters control input value

γ: sway thruster control input value

Dedication

To my parents, for their love and support.

Acknowledgments

I would like to thank Prof. David Lane for, first and foremost, welcoming me into the Ocean Systems Lab to carry out this PhD project. His insights, and especially his ability to ask questions that give one a few weeks' worth of work at least, were invaluable.

I am also in debt to Dr. Emanuele Trucco, who provided me with sound technical advice, and whose coaching on good scientific writing was much appreciated.

A special thanks goes to Dr. François Chaumette, who, during a first visit in Heriot-Watt, and later on, when I came to visit him in France, helped me out with the intricacy of visual servoing. Un grand merci !

My gratitude goes also to Dr. Yvan Petillot as much for his technical help and our brain-storming sessions as for his friendship and his initiation to french wine and food.

Many thanks to Dr. Katia Lebart, for the six months she spent debugging the much hated "Alpha Card" — I fully appreciate how dreadful this has been —, and for all the technical advice given, among other things.

I would also like to thank Dr. Andrea Fusiello for his help with Computer Vision while he was visiting Heriot-Watt. His clear explanations were invaluable.

A very special thanks goes to Joe, a.k.a. "Don Ioseba Tena Ruiz", without whom I would not have been able to write up. Our coffee breaks discussions did improve the contents of our respective thesis, or so I hope.

I had a great time working in the Ocean Systems Lab during all those years: thanks to Adriana, Esther, Francesco, Costas, Nicolas, Kelvin, Len, Francesca, and Geoff for great times.

Finally, although it might not be obvious, this thesis would not have been such an enjoyable period in my life without the Aikido gang, as much for Aikido training as for after-training socializing.

Abstract

Conventional underwater sensors are not well suited to the task of aiding unmanned underwater vehicles to hover. These sensors suffer from several drawbacks such as low sampling rates, low resolution, complexity of operations, drift and cost. Underwater video cameras, however, can provide local measurements of position with respect to a local object. Underwater vision presents several challenges: it suffers from a limited range and poor visibility conditions. Besides, recovering motion from images requires high computing power, which is a limited resource on-board most underwater vehicles.

The main objective of this thesis was therefore to investigate visual control methods to dynamically position a typical underwater vehicle with respect to a fixed object. These methods also had to have computing power's needs compatible with off-the-shelf embedded computers that can be operated on real vehicles.

A hybrid visual servoing technique, adaptated from the 2 1/2 D visual servoing scheme, was proposed. Its performance was assessed in simulations using a six degrees-of-freedom nonlinear dynamic model of an Remotely Operated Vehicle. The effects of sea current disturbances, the target's orientation, and noise under sparse feature tracking conditions was studied. The proposed method proved stable and took into account the restrictive controllability of the vehicle.

A 2-D visual servoing scheme which employed the Shi-Tomasi-Kanade sparse feature tracker on unmarked planar targets in a water tank was then proposed. The scheme controlled a planar Cartesian robot which emulated the dynamic behaviour of the surge and sway degrees-of-freedom of a typical underwater vehicle. The effect of sea current disturbances on the stability and performance of the control scheme was also studied.

An underwater experimental evaluation of the Shi-Tomasi-Kanade feature tracker under various conditions of lighting and relative speed between the underwater scene and the camera was also performed.

Chapter 1

Introduction

Unmanned Underwater Vehicles or UUVs, are widely used for sea exploration and exploitation. Their applications are numerous: inspection and repair of oil and gas facilities, cable-laying for telecommunications and scientific studies of the deep ocean are but a few. The majority of these applications requires the ability to *dynamically position* the UUV with respect to an object, whether it be a cable, an underwater structure, a sunken ship or the seabed. The dynamic positioning of an UUV can also be seen as a prerequisite to docking tasks when the vehicle has to direct itself and dock onto a seabed structure. This capability, however, poses several challenges.

Unlike terrestrial mobile robots, floating robots move in six degrees-of-freedom, and are subject to constant and unknown disturbances caused by sea currents. Tethered UUVs, commonly termed Remotely Operated Vehicles or ROVs, are also subject to dynamic disturbances due to the drag of their umbilical, which can reach several kilometres in length! Hydrodynamic characteristics of UUVs contribute to make them nonlinear time-varying systems. Often, their degrees-of-freedom are coupled, adding to the motion control problem. Dynamic positioning of UUVs is therefore not trivial.

One of the requirements of a motion control system is the ability to obtain position or velocity measurements from sensors. Sensing techniques in water are predominantly acoustic since electromagnetic waves rapidly attenuate in sea water. There is an obvious trade-off between range and positioning accuracy: the lower the frequency, the further the sensor can "sense", and the worse the positioning accuracy becomes. Acoustic sensors hardly provide the sub-metre accuracy required by dynamic positioning. They also suffer from limited sampling rates, rendering the control problem even more challenging. Other conventional non-acoustic sensors such as compasses, or inclinometers, can be biased, and when they are integrating devices, exhibit drift

properties due to measurement noise. They also have limited sampling rates.

Another downfall of conventional underwater sensors is that they generally provide a measurement relative to an arbitrary datum, such as latitude and longitude. Visual methods, however, can provide measurements relative to a *local* object. For example, when inspecting a ship's wreck, a video camera would produce information on the relative position of the wreck with respect to the camera, whereas a long baseline triangulation system would supply an absolute position unrelated to the ship's location. This fact motivated the investigation of visual techniques to station keep an UUV with respect to a local target.

This thesis tackles the issue of dynamically position or *station-keep* an underwater vehicle with respect to a fixed target by means of a single on-board video camera. The visual control methods investigated in this thesis should also be easily implemented on actual UUVs. In particular, Autonomous Underwater Vehicles or UUVs are equipped with typical and affordable state of the art embedded computers (such as PC104) and their computing power is limited. Therefore, sophisticated image processing techniques, however promising, were not investigated because they were too computationally expensive. Consequently the thesis was restricted to linear computer vision techniques to meet this computational cost requirement.

As shall be seen in the following chapters, despite using linear computer vision techniques, the state of the art embedded computer used restricted the visual control sampling rate to 5 Hz. In order to validate the visual control techniques presented in this thesis, a Cartesian robot emulated the **typical motion behaviour** of an underwater robot. The emulation was based on an experimental nonlinear dynamic model of the ROV ANGUS previously operated by the Ocean Systems Laboratory, and no longer in use. When it was operated, its controllers ran at 50 Hz. One of the challenges of this thesis was to design a stable visual controller at a much slower control rate of 5 Hz for this nonlinear time-varying system.

The context and the motivation of this thesis' work has been briefly introduced. The chapter now offers a review of visual control methods for underwater vehicles' dynamic positioning. The challenges of underwater vision will also be highlighted. This

thesis' contribution to the field will be presented, and the thesis' plan announced.

1.1 Visual control of robots

The idea of using visual sensors such as cameras to perform robotic tasks is far from new. The first attempts took place in the early 70's, and employed a "look and move" open-loop strategy [76]. The resulting positioning accuracy of the robot was therefore directly dependent on both the accuracy of the visual sensor and the robot end-effector. A natural idea to further increase the system's positioning accuracy was to "close the loop" around the visual sensor. This is referred to in the literature as *visual servoing*, a term that will be used in the remaining of this thesis:

> "the task in visual servoing is to use visual information to control the position and orientation or *pose* of a robot's end-effector with respect to a target object or a set of target features. For a mobile robot, the visual servoing task is to control the vehicle's pose (position and attitude) with respect to some visual landmarks." (definition adapted from [39])

In this thesis, the focus has been put on closed-loop visual control or visual servoing, since the open-loop approach lacks robustness. This section is organised as follows: first, a taxonomy of visual servoing techniques, adapted from [9] is presented, then a review of visual servoing methods applied to underwater vehicles is given.

1.1.1 Taxonomy

A major classification of visual servoing strategies distinguishes between *position-based* (or 3-D) visual servoing and *image-based* (or 2-D) visual servoing. Recently, hybrid approaches have also been proposed [16, 50]. This classification is based on the nature of the feedback information used in the robot control loop.

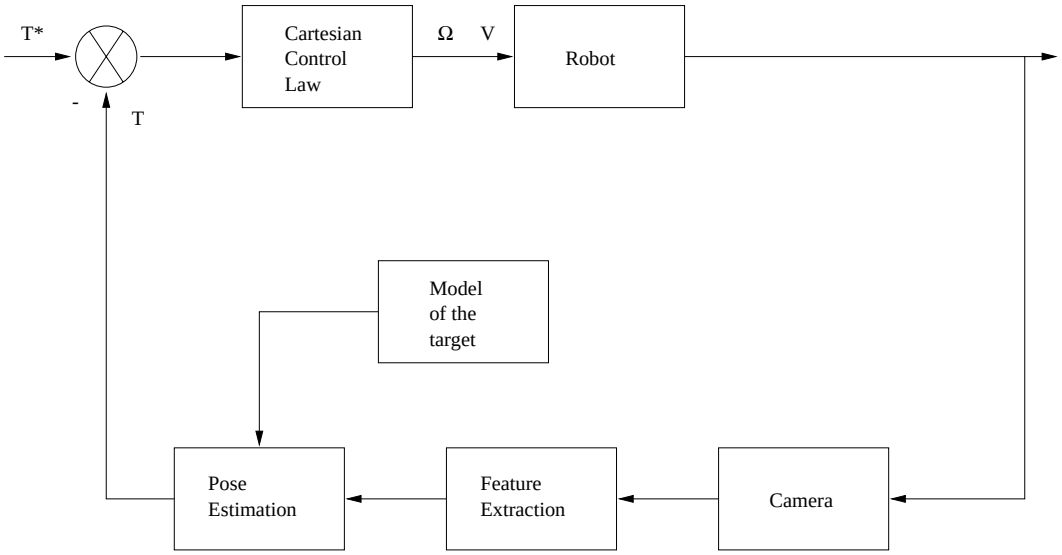


Figure 1.1: 3-D visual servoing: the rigid frame transformation $\mathbf{T}$ or pose between the target and the camera is measured at each cycle and compared with the desired pose $\mathbf{T}^*$. A classic Cartesian controller can be used in the robot control loop.

Position-based visual servoing

In position-based visual servoing [7, 91, 92] (also referred to as *3-D visual servoing*) features are extracted from the image, and used to compute the target's pose[1] with respect to the camera (see figure 1.1). The error signal fed into the robot's position controller is the difference between the desired pose $\mathbf{T}^*$ and the estimated pose $\mathbf{T}$ of the target with respect to the camera.

The main advantages of this type of approach are to allow the use of existing Cartesian controllers by the robots, and to separate the issue of pose estimation from image data from the computation of the feedback signal. The camera trajectory is directly controlled in Cartesian space.

Nevertheless, 3-D visual servoing raises several difficulties. Since the robot's trajectory defined by the control is in Cartesian space, there is no control in the image, and therefore no guarantee that the target will stay in the field of view during the servoing task. In addition, the pose estimation problem from visual data is not trivial. In chapter 4, a few linear methods to compute the relative pose of an object with respect to a camera will be presented. Linear methods are sensitive to noise, and nonlinear ones are too computationally expensive to be of use for visual servoing

[1] *3-D* transformation (translation and rotation) relating the target to the camera.

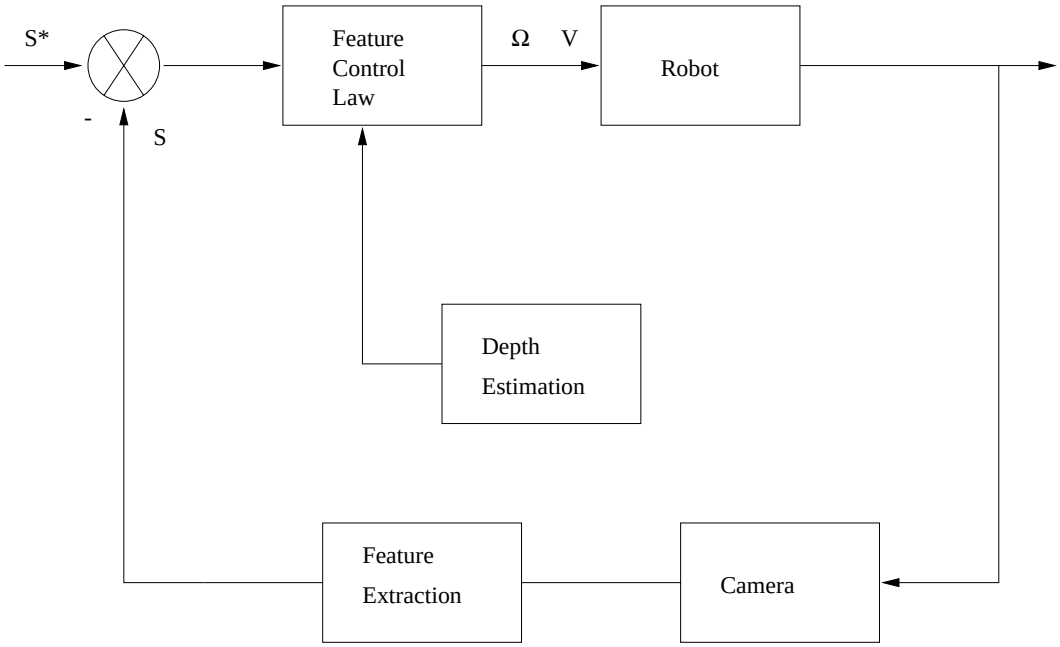


Figure 1.2: 2-D visual servoing.

applications. Other pose estimation methods are based on a geometric model of a target, for example, assuming the object is an ellipse, or a square. This restricts greatly the potential applications of visual servoing. This family of techniques is unsuitable for use in a "natural" and unstructured environments. Camera calibration errors or modelling inaccuracies of the target may be the cause of wrong estimates between the current and desired poses, and the servoing might not converge. Finally, the stability of this type of system has proved, so far, impossible to solve analytically [6].

To render the control more robust and to stop the visual target from leaving the field of view, researchers have investigated visual servoing techniques based solely on visual information. These techniques are termed *image-based visual servoing*.

Image-based visual servoing

In image-based control, the pose estimation is solved implicitly: when the current view of the object matches its desired view, then the camera has reached its desired configuration [9]. More formally, let $\mathbf{r}$ represent the coordinates of the centre of gravity of a mobile camera in a Cartesian reference frame. Let $\mathbf{s}$ be a vector of image parameters, and $\dot{\mathbf{s}}$ represent the derivative of $\mathbf{s}$ with respect to time. Image-

based visual servoing introduced a linear mapping relating $\dot{\mathbf{s}}$ and $\dot{\mathbf{r}}$:

$$\dot{\mathbf{s}} = \mathbf{L}(\mathbf{s})\,\dot{\mathbf{r}}. \tag{1.1}$$

$\mathbf{L}(\mathbf{s})$ is called the *interaction matrix* or *image Jacobian matrix*, introduced the first time by Weiss [88]. Image features can be of diverse nature: image point co-ordinates, segment length and orientation, ellipse parameters, etc. Chaumette derived expressions for the interaction matrix for common image features such as point co-ordinates, ellipses, lines, cylinders [5]. The most common interaction matrix is based on the motion of points in the image ([10, 20, 51, 79] are but a few examples). Assume that the camera's velocity vector, with respect to a fixed camera frame, is $\dot{\mathbf{r}} = [\mathbf{V}^T, \mathbf{\Omega}^T]^T$. Let $\mathbf{M} = [x, y, z]^T$ be a 3-D point belonging to a rigid object expressed in the camera reference frame, and let $\mathbf{s} = [u, v]^T$ be the image plane co-ordinates of $\mathbf{M}$. The velocity of $\mathbf{M}$ expressed in the camera reference frame is given by:

$$\dot{\mathbf{M}} = \mathbf{\Omega} \times \mathbf{M} + \mathbf{V} \tag{1.2}$$

The interaction matrix relating to this point is given by:

$$\mathbf{L}(\mathbf{s}) = \begin{bmatrix} f/z & 0 & -u/z & -uv/f & (f^2+u^2)/f & -v \\ 0 & f/z & -v/f & (-f^2-u^2)/f & uv/f & u \end{bmatrix} \tag{1.3}$$

where f is the focal length of the camera. The interaction matrix relates image-plane velocity $\dot{\mathbf{s}}$ of a point to its relative velocity with respect to the camera $\dot{\mathbf{M}}$ so that:

$$\dot{\mathbf{s}} = \mathbf{L}(\mathbf{s})\,\dot{\mathbf{M}}. \tag{1.4}$$

Derivations of $\mathbf{L}$ can be found in a number of references including [20, 39].

The simplest approach to image-based visual servoing is to use an interaction matrix computed from at least four points of which no three of them are collinear. For instance, four 3-D points forming a square make an adequate visual target (see [5]).

The interaction matrix is then built by stacking up the 2×6 interaction matrices of equation 1.3 with each image point. If the interaction matrix thus built is non-singular, one can use its pseudo-inverse $\mathbf{L}(\mathbf{s})^+$ to set a desired camera's speed $\dot{\mathbf{r}}^*$[2]:

$$\dot{\mathbf{r}}^* = \mathbf{L}(\mathbf{s})^+ \dot{\mathbf{s}}. \tag{1.5}$$

However, more sophisticated approaches have been proposed, notably [20] where the task function paradigm is used [72]. In this thesis, the task function concepts were employed. As an example, consider that the desired position of the camera is reached when, in the image, the current $\mathbf{s}$ and the desired $\mathbf{s}^*$ feature vectors coincide. The vision-based task can be defined as the regulation to zero of function $\mathbf{e}$:

$$\mathbf{e} = \mathbf{s} - \mathbf{s}^*. \tag{1.6}$$

$\mathbf{e}$ is called the *task function*. Now, using eq. 1.1, the interaction matrix relates the time derivative of $\mathbf{e}$ with respect to the camera velocity:

$$\dot{\mathbf{e}} = \dot{\mathbf{s}} = \mathbf{L}(\mathbf{s})\,\dot{\mathbf{r}}. \tag{1.7}$$

If one wishes an exponential decrease of the task function towards zero such that:

$$\dot{\mathbf{e}} = -\lambda\,\mathbf{e} \tag{1.8}$$

where λ is a positive scalar, the control law is given by:

$$\dot{\mathbf{r}}^* = -\lambda\,\mathbf{L}^+(\mathbf{s})\mathbf{e}. \tag{1.9}$$

where $\dot{\mathbf{r}}^*$ is the desired camera velocity. A functional diagram of 2-D visual servoing is pictured in figure 1.2.

It has been demonstrated that 2-D visual servoing strategies are less sensitive to camera calibration errors than 3-D visual servoing schemes [19]. Since the control is specified in the image plane, the camera's trajectory can be erratic. Besides, the

[2]The superscript "+" denotes the pseudo-inverse of a matrix.

interaction matrix is dependent on the estimation of the imaged point depth, z. Its value can be estimated by a classic pose estimation algorithm, thus presenting the same drawbacks as in position-based visual servoing techniques. Adaptive methods to compute depth have also been explored [80], or direct on-line methods to estimate the interaction matrix combining information about the robot and the image motion [65]. The interaction matrix may not be of full rank, therefore image-based visual servoing may be unstable when the computation of the pseudo-inverse of the interaction matrix is close to a singularity [6]. These singularities have proved, so far, impossible to compute analytically.

Hybrid methods

In an attempt to combine advantages from both position-based and image-based visual servoing approaches, *hybrid* methods have been designed [16, 50, 97]. The control task is defined partially in the image, and partially in Cartesian space. In [50], image-based visual servoing is used to control translational degrees-of-freedom, while the epipolar geometry[3] between the current and the desired images is employed to estimate the rotational degrees-of-freedom. This method, termed *2 1/2 D visual servoing*, has the property of decoupling the control of the axes of rotation from the control of the axes of translation. A complete mathematical treatment will be given in chapter 6. A similar approach was proposed in [16]. The principle of the 2 1/2 D visual servoing technique is depicted in figure 1.3. The translational degrees-of-freedom parallel to the image plane, and the remaining translation along the optical axis of the camera are estimated and fed into a first controller. The full rotation (axis $\mathbf{u}$ and angle θ) is estimated and fed into a controller constraining the rotational degrees-of-freedom.

1.1.2 Visual control of underwater vehicles

So far, in the underwater robotics field, few attempts have been made to use vision sensors for control [52, 44, 69, 61]. One of the main reason is that underwater vision

[3]See chapter 4 for an account of epipolar geometry.

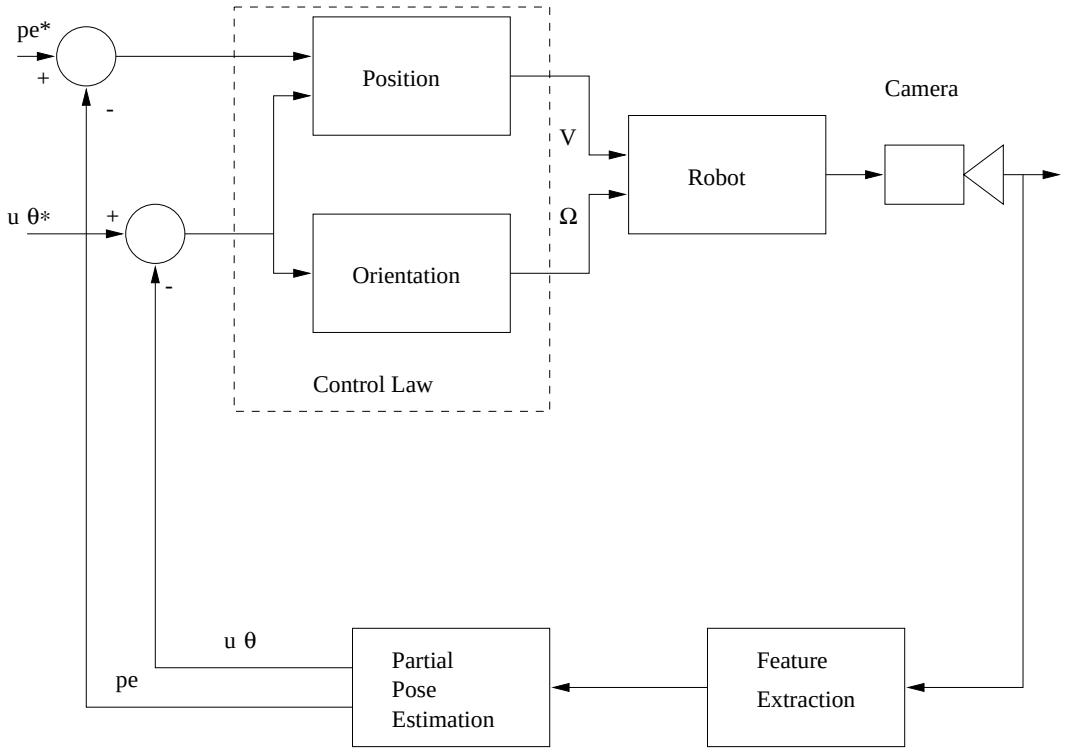


Figure 1.3: A hybrid visual servoing method: the 2-D 1/2 approach.

presents several challenges. Due to the properties of water, optical waves are rapidly attenuated. The range of a video camera is typically between two and ten metres. Backscattering caused by marine snow, that is the presence of floating organic or inorganic particles in water which reflect light, degrades the visibility conditions. Underwater cameras have a comparatively high sampling rate (video rate is 25 Hz) and their resolution is better than a conventional underwater sensor. For example, a pixel of a CCD array of a camera one metre away from a target can represent 1 mm in the real world. However, video information is rich and requires high processing power.

Dynamic positioning has been studied by Marks *et al.* [52]. Their method employed a stereo camera attached to the MBARI's OTTER vehicle. Laplacian of Gaussian filtering, and image correlation were performed with dedicated hardware. Assuming that the vehicle was stable in roll and pitch, and that depth information was available by means of another sensor, they could estimate the motion on the remaining degrees-of-freedom. Although results of motion estimation were given and successful station-keeping was demonstrated, no quantitative performance was available due to

a lack of an external ground truth measuring device. This method was later transferred onto the MBARI Ventana ROV, and at sea dynamic positioning results were given. Proportional-Derivative (PD) linear controllers were employed and again, no external positioning device was used to provide ground truth. Errors were measured in the image in pixels.

Negahdaripour *et al.* proposed a different approach to solve the station-keeping problem [61]. Motion information was estimated with a single camera by computing the optical flow between consecutive images. Optical flow is basically an approximation of the 2-D motion field, i.e. the projection of the 3-D motion field of objects in the image plane. The camera was rigidly fixed to a three-thruster free floating vehicle, and was made to look towards the bottom of a water tank. The motion of the camera, up to a scale factor, was extracted from the optical flow, and fed into Proportional Integral Derivative (PID) position controllers. Heading, and horizontal degrees-of-freedom were controlled. Since the computation of optical flow is expensive, images were reduced in size to provide a sample rate of 30 images per second.

Rives and Borelly used the task function approach [72] to perform pipe-following with the IFREMER VORTEX ROV in their swimming pool [69]. The edges of the pipes were extracted and tracked in user-selected windows. The tracking was improved by Kalman filtering the feature parameters ((ρ, θ) polar representation of segments).

Finally, a related approach, closer to the *active vision* field, was proposed by Crétual *et al.* [11]. A pan and tilt camera performed a stabilisation task on underwater images. The 2-D motion was estimated with a multi-resolution optical flow method called RMR [63], and a task function was devised to constrain the pan and tilt of the camera. Dry validation was carried out on underwater videos projected on a TV screen. The camera was able to track successfully a region in the image sequence.

1.2 The proposed approach

The objective of this thesis was to investigate state of the art visual servoing algorithms to station-keep typical UUVs. The computing power required to run these algorithms should be compatible with the state of the art embedded computers used on Autonomous Underwater Vehicles (AUVs). The main contribution of this thesis was to put together known computer vision and visual control techniques in a novel difficult application: underwater vehicle dynamic positioning. Therefore, a special emphasis was placed on the characterisation of each component belonging to the visual control scheme throughout the chapters.

The visual dynamic positioning techniques proposed in this thesis distinguished themselves from the previously published methods described in section 1.1.2 in several aspects. Contrary to the work of Marks *et al.* [52], which used a stereo system with dedicated hardware, a single camera was employed with an off-the-shelf embedded computer. In addition, the vehicle's motion was not estimated from the optical flow as in [61], but was based on the ability to track sparse feature points in the acquired images of an underwater "natural" scene.[4]

The first method proposed by the thesis can be classified as a hybrid visual servoing approach. It is an extension of the 2 1/2 D visual servoing technique, specifically adapted to an underactuated ROV. Its validity is demonstrated in simulation on a six-degrees-of-freedom model of an ROV. It is able to constrain the four controllable d.o.f. of the ROV, that is: heading, surge, sway and heave.

The second approach is a more classic 2-D visual servoing technique designed to perform the dynamic positioning in the horizontal d.o.f. of the ROV. The technique was implemented on a Cartesian robot moving in a water tank. The surge and sway dynamics of an ROV were emulated on the Cartesian robot, mixing simulation components with real hardware. The Cartesian robot was therefore animated with a motion which closely replicated the dynamics of a typical underwater vehicle.

[4]By "natural", it is meant that the scene was not in any way, specifically designed to suit the visual servoing scheme, such as for example a set of white disks painted on a black board.

1.3 Thesis' overview

This thesis is, by its nature, multi-disciplinary. Hence a great deal of background theory from the underwater robotic and computer vision fields has to be introduced.

Chapter 2 introduces the main concepts and notation on which underwater vehicle modelling is based. These concepts are then applied to the description of Heriot-Watt University's UUV, ANGUS 002, whose hydrodynamic parameters were experimentally derived [29]. Then, in chapter 3, it is showed how, based on the latter's dynamic model, it was possible to emulate two of its six degrees-of-freedom on a Cartesian robot in a water tank.

Chapter 4 provides the reader with the basic terminology of the computer vision field. In particular, it focuses on a widely used geometric camera model: the pinhole camera, and recalls how, under this camera's model assumption and provided a set of feature correspondences in two images, it is possible to recover the Euclidean 3-D motion of the camera, up to a scale factor.

Recovering Euclidean motion from a pair of images requires the ability to find, in both images, a set of corresponding feature points. In the case of a continuous motion, this correspondence problem has to be solved continuously and is termed *feature tracking*. Chapter 5 deals with this aspect. It describes the feature tracking method used as the basis of this thesis' visual servoing experiments. More importantly, an experimental performance evaluation of the latter is carried out underwater on natural scenes. The tests presented in this thesis are not exhaustive. However, a special emphasis has been put on their design so that they are indicative and realistic of the feature tracking performance in harsh underwater environments.

Chapter 6 presents a solution to the visual station keeping problem based on the 2 1/2 D visual servoing technique developed by Malis *et al.* [50]. It is applied to a simulation of the full ANGUS six degrees-of-freedom dynamic model. It is demonstrated that the approach is well suited to solve the station keeping problem. Its performance is assessed in several adverse conditions. In particular, the thesis explores and analyses the effects on the visual servoing performances due to sea current disturbances, of noise on the feature tracking, and the operation of the

robot above a sloped seabed.

Finally, in chapter 7, a 2-D visual servoing scheme is proposed for the station keeping of a reduced order model of an UUV. This model was emulated onto a 2-d.o.f. Cartesian robot in a water test tank. Real-time experiments results are then presented to demonstrate the validity of the approach in presence of sea current disturbances. These experiments demonstrate that the feature tracking algorithm could be used for visual servoing with natural underwater images.

Chapter 2

Underwater vehicle modelling and control

To investigate the feasibility of the visual station-keeping of UUVs, a nonlinear dynamic model of an ROV was used. Its parameters were experimentally derived in a test tank with a planar motion mechanism [29]. This model [3] was employed both in the simulation studies and in the experiments of this work. The model's behaviour was emulated by a planar Cartesian robot in a test tank.

The main objectives of this chapter are:

- to describe the kinematics and dynamics of an UUV, and
- to derive the associated standard equation of motion.

In this description, special care will be taken to underline the assumptions generally made and the validity of the modelling. From that background, the characteristics of the aforementioned UUV model are presented.

The control problems encountered in UUV guidance and control are then illustrated, as well as how these issues have been tackled by the past research.

2.1 Kinematics of an underwater vehicle

Two main reference frames are necessary to describe the vehicle's motion: an Earth-fixed frame which is assumed to be inertial, and a body-fixed frame, attached to the vehicle. The axes (X, Y, Z) of the body-fixed frame coincide with the principal axes of inertia of the vehicle, and the origin is generally taken at the centre of gravity. The vehicle position, $\boldsymbol{\eta}_p = [x, y, z]^T$, and orientation, $\boldsymbol{\eta}_o = [\phi, \theta, \psi]^T$, (roll, pitch and yaw angles) are then described with respect to the inertial reference frame, while

the linear velocity vector $\boldsymbol{\nu}_p = [u, v, w]^T$ (surge, sway, and heave) and the angular velocity vector $\boldsymbol{\nu}_o = [p, q, r]^T$ (roll rate, pitch rate and yaw rate) are expressed with respect to the body-fixed frame. Figure 2.1 summarises the notation. If $\boldsymbol{\nu} = [\boldsymbol{\nu}_p^T, \boldsymbol{\nu}_o^T]^T$ and $\boldsymbol{\eta} = [\boldsymbol{\eta}_p^T, \boldsymbol{\eta}_o^T]^T$; the vehicle flight path relative to the earth-fixed coordinate system is given by the velocity transformation:

$$\dot{\boldsymbol{\eta}} = \mathbf{J}(\boldsymbol{\eta}_o)\,\boldsymbol{\nu} \tag{2.1}$$

where

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_1 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{J}_2 \end{bmatrix} \tag{2.2}$$

and

$$\mathbf{J}_1(\boldsymbol{\eta}_o) = \begin{bmatrix} c\psi c\theta & -s\psi c\phi + c\psi s\theta s\phi & s\psi s\phi + c\psi c\phi s\theta \\ s\psi c\theta & c\psi c\phi + s\phi s\theta s\psi & -c\psi s\phi + s\theta s\psi c\phi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{2.3}$$

$$\mathbf{J}_2(\boldsymbol{\eta}_o) = \begin{bmatrix} 1 & s\phi t\theta & c\phi t\theta \\ 0 & c\phi & -s\phi \\ 0 & s\phi/c\theta & c\phi/c\theta \end{bmatrix} \tag{2.4}$$

where $s\cdot$, $c\cdot$, and $t\cdot$ stand for the sin(), cos() and tan() functions (e.g. $s\psi = \sin(\psi)$), and the attitude angles are defined so that $\phi \in [0, 2\pi[$, $\theta \in]-\frac{\pi}{2}, \frac{\pi}{2}[$ and $\psi \in [0, 2\pi[$. Note that the velocity transformation matrix $\mathbf{J}$ is termed the *Jacobian matrix* of the robot.

2.2 Dynamics of an underwater vehicle

One way to derive the equations of motion of an underwater vehicle is to apply Euler's axioms (conservation of linear and angular momentum). In this section, the results of this derivation alone will be presented. Most of the material used can be found in [23].

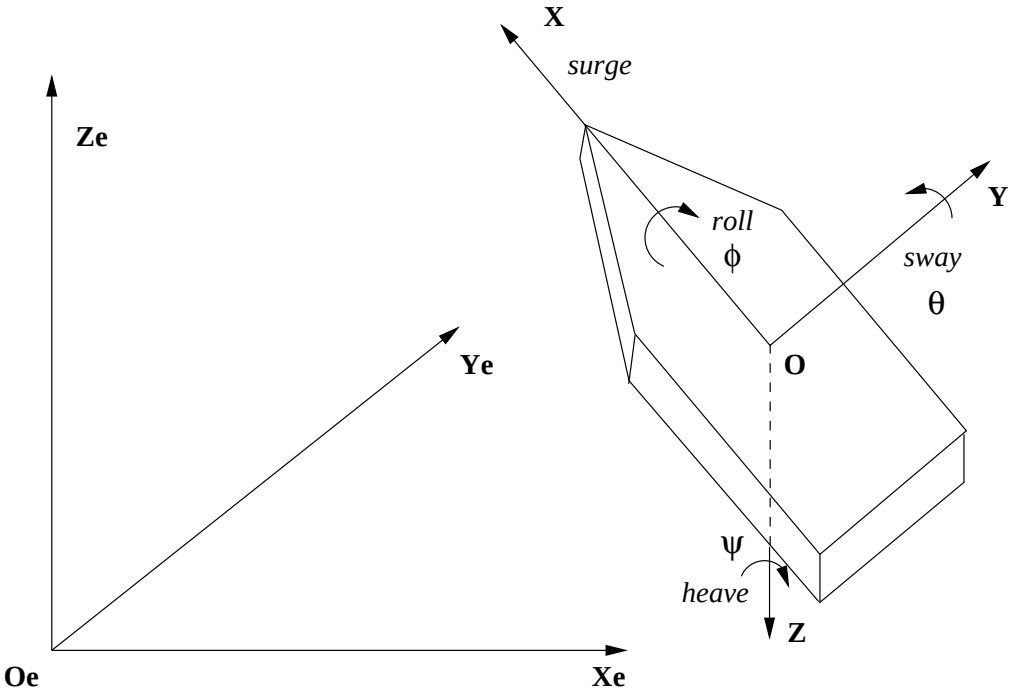


Figure 2.1: UUV co-ordinate frames and motion variables.

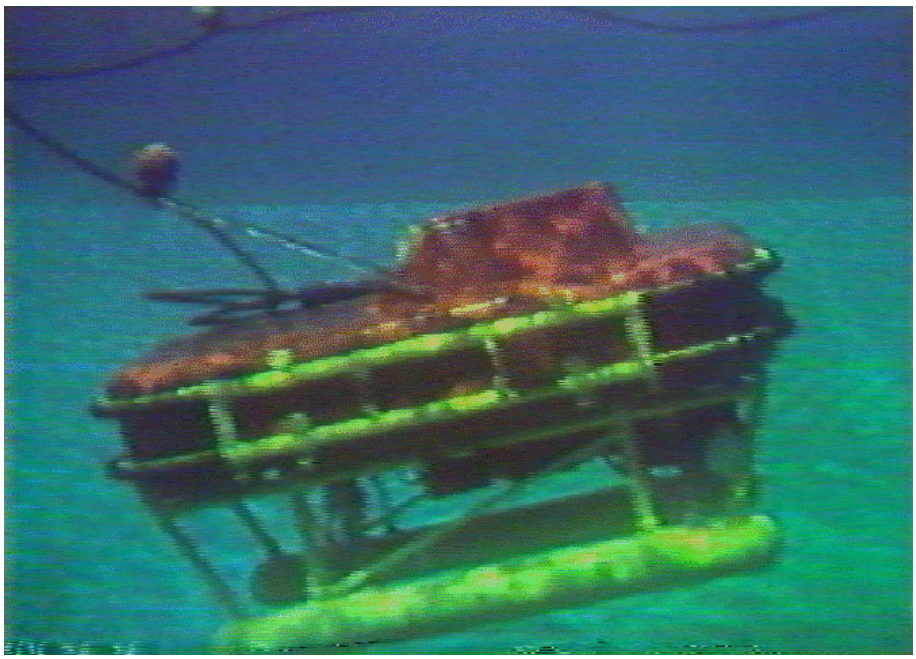


Figure 2.2: ROV ANGUS 002.

According to Euler's axioms, the general six degrees-offreedom (d.o.f.) equations of motion of a rigid body can be expressed as:

$$
\begin{aligned}
m[\dot{u} - vr + wq - x_G(q^2 + r^2) + y_G(pq - \dot{r}) + z_G(pr + \dot{q})] &= X && (2.5)\\
m[\dot{v} - wq + ur - y_G(r^2 + p^2) + z_G(qr - \dot{p}) + x_G(qp + \dot{r})] &= Y && (2.6)\\
m[\dot{w} - uq + vp - z_G(p^2 + q^2) + x_G(rp - \dot{q}) + y_G(rq + \dot{p})] &= Z && (2.7)\\
\mathbf{I}_x\dot{p} + (\mathbf{I}_z - \mathbf{I}_y)qr - (\dot{r} + pq)\mathbf{I}_{xz} + (r^2 - q^2)\mathbf{I}_{yz} + (pr - \dot{q})\mathbf{I}_{xy} & && (2.8)\\
+m[y_G(\dot{w} - uq + vp) - z_G(\dot{v} - wp + ur)] &= K && (2.9)\\
\mathbf{I}_y\dot{q} + (\mathbf{I}_x - \mathbf{I}_z)rp - (\dot{p} + qr)\mathbf{I}_{xy} + (p^2 - r^2)\mathbf{I}_{zx} + (qp - \dot{r})\mathbf{I}_{yz} & && (2.10)\\
+m[z_G(\dot{u} - vr + wq) - x_G(\dot{w} - uq + vp)] &= M && (2.11)\\
\mathbf{I}_z\dot{r} + (\mathbf{I}_y - \mathbf{I}_x)pq - (\dot{q} + rp)\mathbf{I}_{yz} + (q^2 - p^2)\mathbf{I}_{xy} + (rq - \dot{p})\mathbf{I}_{zx} & && (2.12)\\
+m[x_G(\dot{v} - wp + ur) - y_G(\dot{u} - vr + wq)] &= N && (2.13)
\end{aligned}
$$

where m is the dry mass of the vehicle, x_G, y_G, and z_G are the co-ordinates of its centre of gravity expressed in the body-fixed frame, X, Y, Z, K, M, and N are the co-ordinates of the external forces and moments exerted on the vehicle in the same frame. The components of the inertia matrix are $\mathbf{I}_x$, $\mathbf{I}_y$, etc.

Following Fossen's notation [23], these equations can be written in vectorial form as:

$$\boldsymbol{M}_{RB}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_{RB} \tag{2.14}$$

These equations are generally simplified in two ways. First, the centre of gravity $\mathbf{G}$ is taken to coincide with the origin $\mathbf{O}$ of the body-fixed frame, then $x_G = y_G = z_G = 0$. Besides, if some axes of the body-fixed frame are defined so that they coincide with the principal axes of the vehicle, some cross terms of the inertia matrix disappear. For example, if $\mathbf{O}XY$ is a plane of symmetry (ANGUS' case, see figure 2.2), then $\mathbf{I}_{xy} = \mathbf{I}_{yz} = 0$, and the equations read:

$$
\begin{aligned}
m[\dot{u} - vr + wq] &= X && (2.15)\\
m[\dot{v} - wq + ur] &= Y && (2.16)\\
m[\dot{w} - uq + vp] &= Z && (2.17)\\
\mathbf{I}_x\dot{p} + (\mathbf{I}_z - \mathbf{I}_y)qr - (\dot{r} + pq)\mathbf{I}_{zx} &= K && (2.18)\\
\mathbf{I}_y\dot{q} + (\mathbf{I}_x - \mathbf{I}_z)rp + (p^2 - r^2)\mathbf{I}_{zx} &= M && (2.19)\\
\mathbf{I}_z\dot{r} + (\mathbf{I}_y - \mathbf{I}_x)pq + (rq - \dot{p})\mathbf{I}_{zx} &= N && (2.20)
\end{aligned}
$$

Now, one needs to identify the external hydrodynamic forces and moments acting upon the vehicle. For underwater vehicles moving at low speed (to within a few metres per second), the main hydrodynamic forces are:

- added mass and inertia: $\boldsymbol{\tau}_A$
- drag forces: $\boldsymbol{\tau}_D$
- hydrostatic (or restoring) forces: $\mathbf{g}(\boldsymbol{\eta})$
- propulsion forces: $\boldsymbol{\tau}$
- environmental disturbances (sea currents, tether): $\boldsymbol{\tau}_E$

2.2.1 Additional inertial forces

The *additional inertial forces* are pressure-induced forces and moments proportional to the acceleration of the vehicle. The vehicle, while moving, is displacing water, hence creating these forces.

The kinetic energy $\mathbf{T}_A$ of the fluid displaced is given by:

$$\mathbf{T}_A = \frac{1}{2}\boldsymbol{\nu}^T \boldsymbol{M}_A \boldsymbol{\nu} \qquad (2.21)$$

where $\boldsymbol{M}_A$ is the added mass matrix, which is expressed as:

$$\boldsymbol{M}_A = -\begin{bmatrix} X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} \\ Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} \\ Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} \\ K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} \\ M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} \\ N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}} \end{bmatrix} \tag{2.22}$$

The terms of $\boldsymbol{M}_A$ follow the SNAME[1] (1950) notation. For example, the hydrodynamic added mass force Z_A along the z-axis resulting from an acceleration $\dot{v}$ in the y-direction is written as:

$$Z_A = Z_{\dot{v}}\dot{v} \text{ where } Z_{\dot{v}} = \frac{\partial Z}{\partial \dot{v}} \tag{2.23}$$

One can derive the expression of the added mass forces and moments from $\mathbf{T}_A$ by applying *Kirchhoff's equations* (see [23] and references therein). These forces can be summarised as:

$$\boldsymbol{\tau}_A = \mathbf{M}_A\dot{\boldsymbol{\nu}} + \mathbf{C}_A(\boldsymbol{\nu})\boldsymbol{\nu} \tag{2.24}$$

2.2.2 Hydrodynamic damping

Hydrodynamic damping is caused by forced body oscillations, wave drift damping, vortex shedding damping as well as linear and quadratic skin frictions. A body moving at large speeds will present a nonlinear and coupled damping term $\boldsymbol{\tau}_D = \mathbf{D}(\boldsymbol{\nu})\boldsymbol{\nu}$. However, for an ROV or for an AUV, it is a good approximation to take into account the linear and quadratic skin frictions alone, so that $\mathbf{D}(\boldsymbol{\nu})$ has a diagonal structure:

$$\begin{aligned} \mathbf{D}(\boldsymbol{\nu}) = &-\text{diag}\{X_u, Y_v, Z_w, K_p, M_q, N_r\} \\ &- \text{diag}\{X_{u|u|}|u|, Y_{v|v|}|v|, Z_{w|w|}|w|, K_{p|p|}|p|, M_{q|q|}|q|, N_{r|r|}|r|\} \end{aligned} \tag{2.25}$$

[1]Society of Naval and Marine Engineering.

where $X_{u|u|} = \frac{\partial X}{\partial(u|u|)}$, etc.

2.2.3 Restoring forces and moments

Restoring forces and moments are hydrostatic forces and moments resulting from the combined action of the buoyancy and the gravity upon the vehicle. The gravitational force $\mathbf{F}_G$ acts through the centre of gravity $\mathbf{G}$ of the vehicle, while the buoyant force $\mathbf{F}_B$ acts through the centre of buoyancy. Let $\mathbf{r}_G = [x_G, y_G, z_G]^T$ and $\mathbf{r}_B = [x_B, y_B, z_B]^T$ be the vectors of the centres of gravity and buoyancy respectively, expressed in the body-fixed frame, then the restoring force and moment (also called *hydrostatic wrench*) $\mathbf{g}(\boldsymbol{\eta})$ is:

$$\mathbf{g}(\boldsymbol{\eta}) = -\begin{bmatrix} \mathbf{F}_G(\boldsymbol{\eta}) + \mathbf{F}_B(\boldsymbol{\eta}) \\ \mathbf{r}_G \times \mathbf{F}_G(\boldsymbol{\eta}) + \mathbf{r}_B \times \mathbf{F}_B(\boldsymbol{\eta}) \end{bmatrix} = \begin{bmatrix} (W-B)s\theta \\ -(W-B)c\theta s\phi \\ -(W-B)c\theta c\phi \\ -(y_G W - y_B B)c\theta c\phi \quad + \quad (z_G W - z_B B)c\theta s\phi \\ (z_G W - z_B B)s\theta \quad + \quad (x_G W - x_B B)c\theta c\phi \\ -(x_G W - x_B B)c\theta s\phi \quad - \quad (y_G W - y_B B)s\theta \end{bmatrix} \tag{2.26}$$

where $W = \|\mathbf{F}_G\|$ and $B = \|\mathbf{F}_B\|$.

2.2.4 Propulsion forces

In the general case, the thrusters' force and moment vector, $\boldsymbol{\tau}$, is a complex nonlinear function $\mathbf{b}$ of the vehicle's velocity vector $\boldsymbol{\nu}$ and the rotational motor speed vector $\mathbf{n} = [n_i]_{i\in[1,p]}$, where p is the number of motors [23]:

$$\boldsymbol{\tau} = \mathbf{b}(\boldsymbol{\nu}, \mathbf{n}). \tag{2.27}$$

Forward and backward thrusts are generally non-symmetrical, unless a special design is implemented. A classical first-order model approximation of the developed thrust T and torque Q for a single screw propeller can be found in [62]. Recent research

work focused on thrusters' modelling is described in [24, 35, 89, 90, 94], and the reader is referred to them for in-depth descriptions. The thrusters which equip the ANGUS ROV will be treated in detail in section 2.3.

2.2.5 Environmental forces

The *environmental forces*, $\boldsymbol{\tau}_E$, are caused by sea currents acting on the vehicle body, and disturbances such as those created by a tether (in the case of an ROV), or a manipulator mounted on the robot. In this thesis, the focus will be put on sea current disturbances only. The hydrodynamic forces and moments depend directly on the velocity of the vehicle with respect to the water surrounding it. Consequently, the sea current velocity must be involved in the equations of motion. It is convenient to express the sea current velocity in the world co-ordinate system.

Let $\dot{X}_c$, $\dot{Y}_c$ and $\dot{Z}_c$ be the components of the sea current velocity in the world (inertial) reference frame, and u_c, v_c and w_c its components in the body-fixed frame. The transformation between the inertial frame and the body-fixed frame is given by

$$\begin{bmatrix} u_c \\ v_c \\ w_c \end{bmatrix} = \mathbf{J}_1^{-1} \begin{bmatrix} \dot{X}_c \\ \dot{Y}_c \\ \dot{Z}_c \end{bmatrix} = \mathbf{J}_1^{T} \begin{bmatrix} \dot{X}_c \\ \dot{Y}_c \\ \dot{Z}_c \end{bmatrix} \tag{2.28}$$

Let $\tilde{\boldsymbol{\nu}} = [\tilde{u}, \tilde{v}, \tilde{w}, p, q, r]^T$ (where $\tilde{u} = u - u_c$, etc.) be the velocity vector of the submersible's centre of gravity with respect to the water. It is then possible to express the combined actions of the drag forces and the environment forces as:

$$\boldsymbol{\tau}_D + \boldsymbol{\tau}_E = \mathbf{D}(\tilde{\boldsymbol{\nu}})\tilde{\boldsymbol{\nu}} \tag{2.29}$$

Now, taking into account the added mass effects and the environmental disturbances, it is possible to express eq. (2.14) in a more compact form:

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\tilde{\boldsymbol{\nu}})\tilde{\boldsymbol{\nu}} + \mathbf{g}(\boldsymbol{\eta}) = \boldsymbol{\tau} \tag{2.30}$$

where

$$
\begin{aligned}
\boldsymbol{M} &= \boldsymbol{M}_{RB} + \boldsymbol{M}_{A} \\
\boldsymbol{D} &= \boldsymbol{D}_{RB} \\
\boldsymbol{C}(\boldsymbol{\nu}) &= \boldsymbol{C}_{RB}(\boldsymbol{\nu}) + \boldsymbol{C}_{A}(\boldsymbol{\nu})
\end{aligned}
$$

and the relation between $\boldsymbol{\eta}$ and $\boldsymbol{\nu}$ is given by eq. (2.1).

2.3 ANGUS 003 dynamic model

ANGUS 003 is a work-class ROV previously built and characterised in-house by the Ocean Systems Laboratory of Heriot-Watt University. Although no longer used for experiments, a nonlinear dynamic model has been experimentally identified in a test tank with a planar motion mechanism [3]. This vehicle possesses two independent back thrusters (*port* and *starboard*) for forward motion and heading motion (using differential thruster values to rotate). ANGUS also has four vertical thrusters for heave motion, and one side thruster for sway. The vertical thrusters cannot be controlled independently, therefore roll and pitch motions are not controllable. Figure 2.3 illustrates the thruster configuration. In the following paragraphs, a mathematical expression of the dynamic model is described. Numerical values are given in Appendix A.

The dynamic model of the vehicle can be expressed as:

$$
\mathbf{M}\,\dot{\boldsymbol{\nu}} = \mathbf{B}\begin{bmatrix}
\tilde{u}(|\tilde{u}| + D_u) \\
\tilde{v}(|\tilde{v}| + D_v) \\
\tilde{w}(|\tilde{w}| + D_w) \\
p(|p| + D_p) \\
q(|q| + D_q) \\
r(|r| + D_r)
\end{bmatrix} + \mathbf{C}(\boldsymbol{\nu}) + \mathbf{g}(\boldsymbol{\eta}) + \mathbf{E}(\boldsymbol{\nu})\,\mathbf{U}_{\tau} + \mathbf{F}(\boldsymbol{\nu})\,|\mathbf{U}_{\tau}| \tag{2.31}
$$

where $\mathbf{M}$ is a 6×6 mass matrix as defined by eq. (2.30), $\mathbf{B}$ is a 6×6 drag matrix and $\mathbf{D} = [D_u, D_v, D_w, D_p, D_q, D_r]^T$ is a 6×1 laminar flow drag vector. The hydrostatic wrench $\mathbf{g}$ in ANGUS case has been shown to be [18]:

$$\mathbf{g}(\boldsymbol{\eta}) = \begin{bmatrix} (B-W)s\theta \\ (W-B)c\theta s\phi \\ (W-B)c\theta c\phi \\ -HBc\theta s\phi \\ -HBs\theta \\ 0 \end{bmatrix} \tag{2.32}$$

where H is the *metacentric height*, i.e. the height of the centre of buoyancy above the centre of gravity.

Since the propellers are less efficient when turning in reverse, the action of the thrusters has been encoded in the two 6×4 thrust matrices $\mathbf{E}$ and $\mathbf{F}$. The thrust forces and moments vector, $\boldsymbol{\tau}$, is then expressed as:

$$\boldsymbol{\tau} = \mathbf{E}(\boldsymbol{\nu})\,\mathbf{U}_\tau + \mathbf{F}(\boldsymbol{\nu})\,|\mathbf{U}_\tau| = \mathbf{E}(\boldsymbol{\nu}) \begin{bmatrix} port \\ starboard \\ \gamma \\ \alpha \end{bmatrix} + \mathbf{F}(\boldsymbol{\nu}) \begin{bmatrix} |port| \\ |starboard| \\ |\gamma| \\ |\alpha| \end{bmatrix} \tag{2.33}$$

where $\mathbf{U}_\tau$ is a normalised control vector whose components' value ranges from -100 % to $+100$ %. The control parameter of the lateral thruster is γ, while α is the control parameter of the four vertical thrusters. As their names suggest, *port* and *starboard* are the control parameters of the port and starboard backward thrusters respectively.

The vector of centrifugal forces $\mathbf{C}(\boldsymbol{\nu})$ has been greatly simplified because of the lack of knowledge concerning the values of most of its elements. The basic centrifugal forces, applied by the virtual inertias of the vehicle, have been taken into account by means of the following vector:

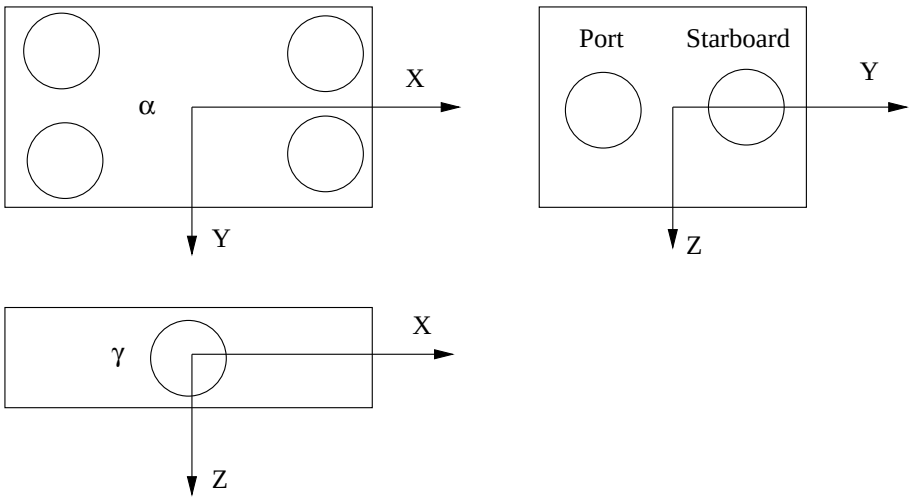


Figure 2.3: ANGUS 003 thrusters' configuration.

$$\mathbf{C}(\boldsymbol{\nu}) = \begin{bmatrix} \mathbf{M}_{11}(vr - wq) \\ \mathbf{M}_{22}(wq - ur) \\ \mathbf{M}_{33}(uq - vp) \\ (\mathbf{M}_{55} - \mathbf{M}_{66})qr - \mathbf{M}_{46}pq \\ (\mathbf{M}_{66} - \mathbf{M}_{44})pr + \mathbf{M}_{46}(p^2 - r^2) \\ (\mathbf{M}_{44} - \mathbf{M}_{55})pq + \mathbf{M}_{64}qr \end{bmatrix} \tag{2.34}$$

This model, although simplified, is highly nonlinear in many aspects. The thrusters' efficiency is different depending on the propellers' direction of rotation, and of the vehicle's speed as well. Cross-coupling terms between axes are another element of nonlinearity. Finally, one of the elements of the drag matrix $\mathbf{B}$ depends on the sign of the surge speed u. Figure 2.4 is a functional diagram of the ANGUS model: the co-ordinate transformations, the action of a sea current and of the control inputs are depicted.

2.4 Control of underwater vehicles

In this section, a brief overview of the research in underwater vehicles' control will be given. In practice, PI or PID linear controllers are commonly used on ROVs (see e.g. [42]). They have the advantage of simplicity but require high gains to achieve the desired performances such as small steady-state and tracking errors.

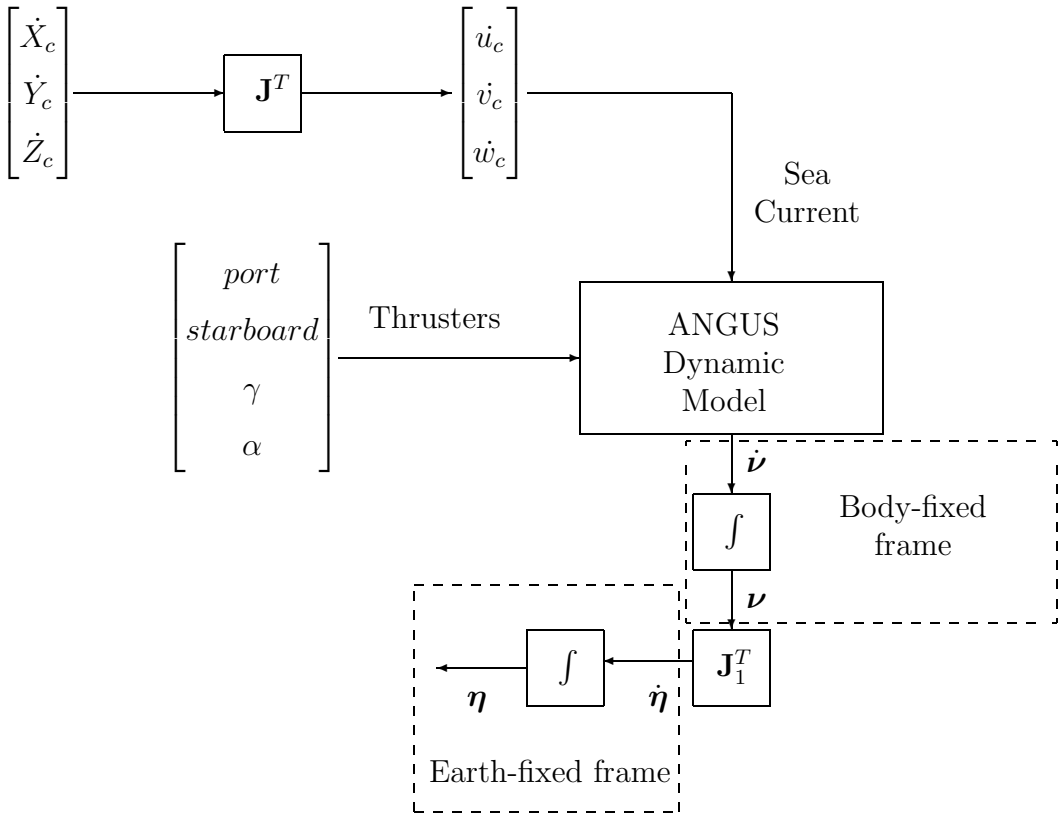


Figure 2.4: Block diagram of the dynamic model of ANGUS 003.

Furthermore, their performances are only valid for a given configuration. To tune a linear controller, the common method is to linearise the model of the robot around an *operating point*. For example, a cruise controller at 1 knot would require the nonlinear model of the UUV to be linearised around a surge speed of 1 knot, and all other speeds set to zero. Therefore, in theory, as many controllers as operating modes will be required, which is not practical. In addition, ROVs' dynamics are nonlinear, coupled and do not present privileged directions of operation. As a result, a cruise controller and a heave controller would not be independent if the surge and heave degrees-of-freedom were coupled. For streamlined AUVs, it is however possible to design controllers for privileged directions such as in the work of [34] where the authors assume lightly interacting degrees-of-freedom. Whereas, as stressed in [31], when significant changes in the vehicle's dynamics occur, conventional controllers with fixed gains fail to guarantee the high quality response of the overall system. This fact lead researchers to investigate nonlinear control methods [41].

A solution that takes into account the nonlinearities of underwater vehicles applies model-based control schemes such as gain scheduling [4], pole cancellation [94], or

computed-torque control [102]. However, these schemes assume that the knowledge of the model is accurate. For underwater vehicles, this is not generally the case. The hydrodynamic parameters are at best difficult to evaluate. Some parameters also change depending on the vehicle's payload configuration. For example, consider a scientific ROV which is set to carry out different types of mission, the position and mass of its on-board sensors will change the metacentric height. Alternative control methods have been devised to cater for the dynamic changes in the model's parameters.

Sliding mode control is one such alternative. Yoerger *et al.* applied sliding control to the Jason ROV [95], and in a later publication took into account Jason's thruster dynamics [94]. Refinements to sliding control were proposed by several other researchers. Cristi *et al.* reported an adaptive sliding mode controller combined with a state observer [12]. Healey *et al.* [34] proposed a multivariable sliding mode technique for cruise control at high speeds (10 knots) based on state variable errors, rather than output errors as in [12]. Da Cunha *et al.* proposed a variable structure algorithm requiring only position measurements [13]. Adaptive control approaches have also been investigated by Yuh [96], Fossen and Fjellstad [25], and Fossen and Sagatun [26]. Another interesting approach proposed by Perrier *et al.* [64] combined linear PID controllers with nonlinear PID controllers improving transient responses to heading and depth control.

In addition to the uncertainties in the dynamic parameters, and the unknown disturbances acting on the vehicle such as sea currents and manipulator interactions, the control of an UUV is made even more challenging by slow rate sensors. Conventional underwater sensors such as depthmeters, compasses, and acoustic echo-sounders have an update frequency of a few hertz, hence reducing the available bandwith of the robot. It is then more difficult to reject disturbances of high frequencies. This concludes the overview of underwater vehicle control.

2.5 Conclusions

In this chapter, the necessary theoretical background of underwater vehicles kinetic and dynamic modelling was introduced. The dynamic model of the ROV ANGUS 003 used in the visual servoing experiments of this thesis was then described. This dynamic model was used both in simulation and in real-time experiments. For the latter, it was then necessary to emulate ANGUS' dynamic behaviour on a Cartesian robot. The details of the implementation of this emulation will be presented in the following chapter.

Chapter 3

Emulating ANGUS model on a Cartesian robot

Recall the main objective of this thesis which was "to investigate visual servoing algorithms for the control of underwater vehicles". To meet this objective, two different types of experimental work were carried out.

For the first set of experiments, it was assumed that the image processing part, i.e. the ability to track feature points in underwater images, had been solved. It was then possible to perform closed-loop simulations based on the dynamic model of ANGUS, and to concentrate on the visual task design. These simulations and the visual task design are fully described in chapter 6. These experiments allowed to assess the adequacy of the proposed visual servoing design to control an underwater vehicle, and the performances which could be expected from the design. However, these simulations did not evaluate the solution in underwater conditions with an actual camera sensor and a robot.

To cater for these limitations, a Cartesian robot, placed above a water tank, was used to move a camera underwater (see figure 3.1). In order to emulate real underwater conditions with an UUV, the camera was made to move as if it were attached to the actual ANGUS robot. The dynamic model of ANGUS (previously described in chapter 2) generated the trajectory of the camera. In other words, the ANGUS' dynamic model drove the Cartesian robot. This chapter aims at describing the experimental setup employed so that the camera attached to the Cartesian robot moved as it were mounted on a typical underwater vehicle.

This chapter is structured as follows. First, the mechanical components of the Cartesian robot are described, as well as its control hardware and software architecture, in order to better understand its capabilities and its limitations. Then the method

employed to emulate ANGUS' dynamic characteristics using the Cartesian robot is exposed and the performance of the emulation is assessed.

3.1 The Cartesian robot

The Ocean Systems Laboratory possesses a test tank 4 metres long (X), 3 metres wide (Y) and 2 metres deep (Z) filled with fresh water. A 3-axis Cartesian robot equipped with stepping motors is placed on top of this tank (see figure 3.1). This section describes the work performed to emulate the dynamic behaviour of a subset of ANGUS' axes with this Cartesian robot. This emulated underwater robot will then be equipped with an underwater camera to test and validate in real time visual servoing algorithms. The expected advantage of this approach is the availability of precise positioning information (to within one millimetre accuracy) in the horizontal plane. This kind of accuracy cannot be achieved with standard underwater equipment on UUVs. As such, this experimental setup enabled to assess the positioning performance of the tested algorithms.

The robot was purchased as a complete package from the SIG POSITEC company. The robot originally consisted of three degrees of freedom: translations in the horizontal plane (X, Y), and rotation about the Z axis (θ). The rotational degree of freedom was later converted to a translation in Z.

The communication interface did not allow for the external real-time control of its motion at a sufficient sampling rate for control purposes (2 Hz through a serial line). In addition, the positioning information from the optical encoders was not made available through the communication interface. Therefore, an alternative motion control solution was required to be able to both control the Cartesian robot from an external computer and log the position information of its optical encoders.

The following sections describe in more details the mechanical components of the Cartesian robot, and the control hardware and software used to meet the emulation's requirements.

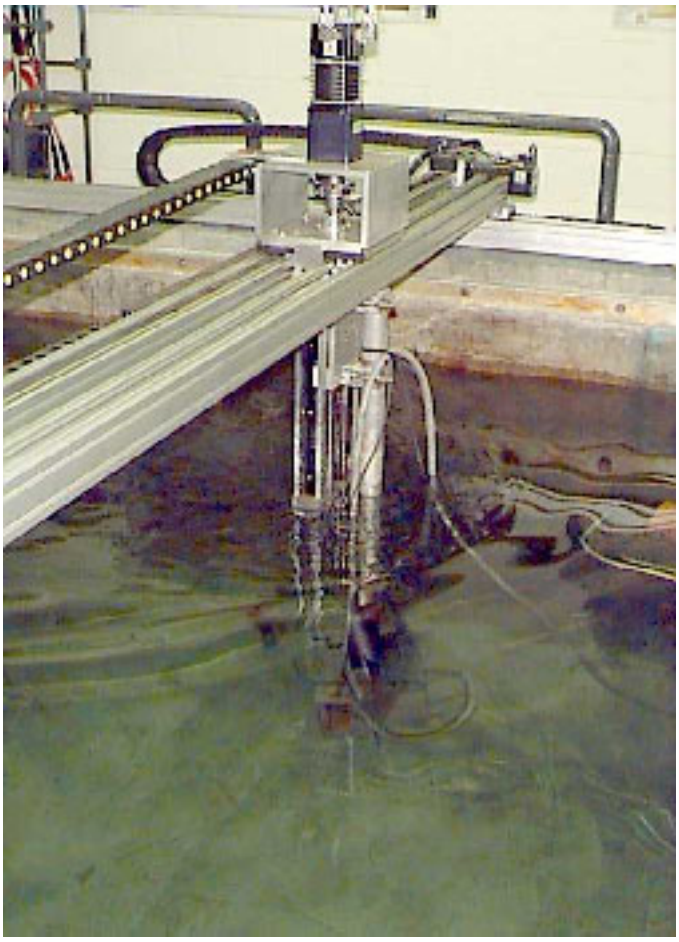


Figure 3.1: The Cartesian robot in the Ocean Systems Lab water tank.

3.1.1 Mechanical positioning components

The Cartesian robot was a BERGHER LAHR portal robot *PR6/2 + H axis* [77] with an added Z axis (see figure 3.1). The X and Y axis were identical, only their stroke differed (3850 mm and 2770 mm respectively). The guide was an aluminium profile, and the rotary motion was transformed into a linear one by a toothed belt, yielding a positioning resolution of 100 mm per revolution.

The Z axis had a 10:1 planetary gear-box (BERGHER LAHR PL50 10:1) driving a lead screw which transformed the rotary motion into a linear one. The resolution of this axis was of 1 mm per revolution. The motion range was approximately 300 mm.

The three 5-phase stepping motors [78] could be operated in two modes: half-step or full step. In full-step mode, the resolution was of 500 steps per revolution, and in half-step mode 1000 steps per revolution. Each motor was further equipped with a 50-500 OTA optical encoder which provided 500 pulses per revolution in each of its two channels which were 90 degrees out of phase.

The overall system's ranges and resolutions are summarised in table 3.1; these figures were taken from [40]. The stepping motors could miss up to 8 steps without it being reported; this explained why the motors' accuracy was equal to 8 times the resolution.

Axis	Stroke (mm)	Resolution (mm/step)	Accuracy (mm)	Repeatability (mm)
X	3,850	0.1	0.8	±0.1
Y	2,770	0.1	0.8	±0.1
Z	300	0.001	0.008	?

Table 3.1: Cartesian robot axes characteristics.

It is clear from this description that the Z-axis could not be used to emulate a vertical motion since it suffered from a reduced stroke, and, as a consequence of its fine resolution, from slow dynamics. In the following section, the hardware and software solution employed to control the Cartesian robot is presented.

3.1.2 Control hardware and software

The motion control card used to servo the stepping motors was a Galil DMC-1360 [30] and was housed in a VME-based real-time system. This card controlled all three axes, and read the optical encoder values. It had several modes of motion, among which two were used: the contouring mode and the jogging mode.

1. The *contouring mode* allowed the user to prescribe an arbitrary position trajectory, and was useful when complex computer-generated trajectories had to be followed. Position increments over a time interval were specified for each axis.

2. In *jogging mode*, each motor ran at a prescribed speed. The user set the jog speed, the acceleration and the deceleration rates. This mode was useful to follow velocity profiles.

The VME crate was a multi-processor computer system designed for real-time applications. It comprised four Motorola processor boards: one of them acted as the host computer and the other three were the real-time targets. The host computer was a MVME167 processor board [55]. It had a SCSI disk drive, a SCSI tape drive and an Ethernet connection. It ran Motorola's UNIX V/68 operating system [59, 60, 57, 58], and was provided with a range of development tools for real-time applications. The target processor boards were MVME162-22 boards and ran the pSOS+m real-time operating system [56]. All processor boards had built-in SCSI, Ethernet Centronics and RS-232 interfaces and connectors. The host computer and the three target boards were sharing the same VME bus, as shown in figure 3.2.

The motion control card had its memory mapped in the VME bus, the software implementation of its driver was such that the shared memory was accessible both from the VME host computer, or from one of the target boards. In other words, it was possible to have a real-time task, running on one of the target boards, controlling the Cartesian robot's motion. This task could also act as a "slave" to an external "master" computer. It could run a visual servoing task, and send commands through the Ethernet connection. These facilities were employed in the implementation of ANGUS' emulation.

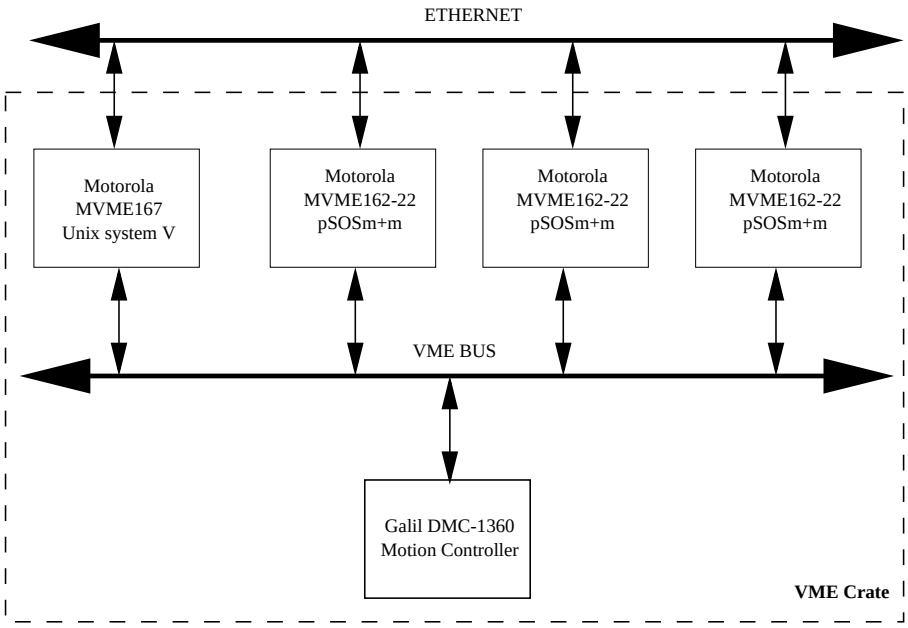


Figure 3.2: VME crate system with host, three targets, and the motion control card.

The following sections present the strategies that have the Cartesian robot emulate the behaviour of the surge and sway d.o.f. of ANGUS.

3.2 Direct control

The first strategy sent the positions to be reached directly to the Cartesian robot as they were made available by the ANGUS model. In this strategy, the Cartesian robot was controlled in contouring mode. Although this approach was simple and made use of the stepping motors in a consistent way, it proved unsatisfactory. Once the model's desired position was reached by the robot, the motion controllers made the robot stop. The desired trajectory was followed, but the desired velocity profile was not. In addition, if the desired trajectory was to be followed within the same time frame as the ANGUS model, the accelerations required for the Cartesian robot to meet this time constraint would have been greater than the actual accelerations of ANGUS. In [40], the author attempted to emulate the dynamic behaviour of ANGUS using the contouring mode of the motion control card. It proved to be impossible without scaling the model by a factor of 6, i.e. 6 cm travelled by the ANGUS model corresponded to 1 cm travelled by the Cartesian robot. This "scaled"

solution was not satisfactory since the goal was to replicate, as best as possible, the actual motion issued from ANGUS' simulation.

Since the contouring mode did not meet the set requirements, the independent jogging mode of the Galil motion card was used instead. In the following section, the details of this alternative strategy shall be described.

3.3 Velocity control of the Cartesian robot

3.3.1 Implementation

To emulate successfully the dynamic behaviour of ANGUS, the jogging mode provided by the motion control card had to be used. The implementation had to meet two main requirements:

1. the update rate between thruster values had to be the same since the visual servoing sampling time T_{vs} was 200 ms because of available computer power;
2. in addition, the velocity profile of the ANGUS model had to be replicated by the Cartesian robot in the same time frame. Both velocity profiles, the model's and the robot's, should be as close to one another as possible.

In order to meet these requirements, the dynamic model code and the jogging command ran on one of the real-time target boards (overall execution time: $T_{robot} = 6.5$ ms) 30 times with the same thruster values to reach a $T_{vs} = 200$ ms cycle time. In addition, it was necessary to implement a simple PI velocity control between the ANGUS model and the Cartesian robot to reduce the tracking errors observed on the velocity profiles. In the following sections, $X_p(k)$ and $Y_p(k)$ denote the positions on the X- and the Y-axis of the Cartesian robot respectively. These positions were measured by the optical encoders of the stepping motors. Similarly, K_{ix} and K_{iy} denote the integration coefficients on each axis. The details of the implementation are shown in figure 3.3 for the X-axis only.

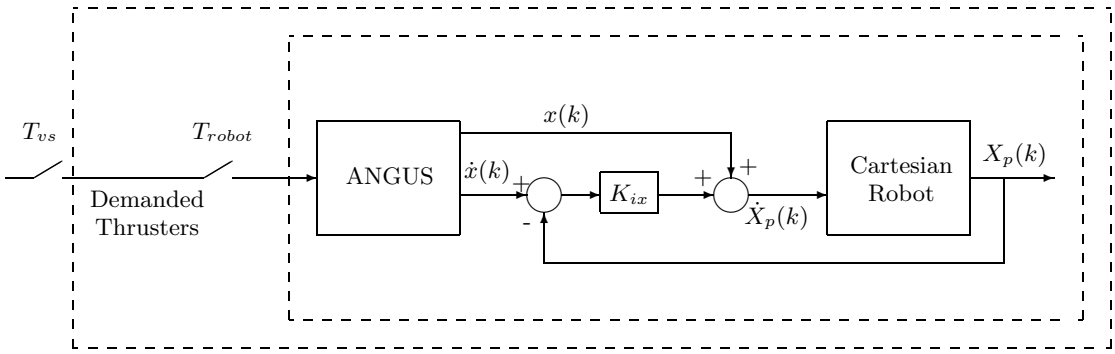


Figure 3.3: Software architecture and implementation of ANGUS emulation. $T_{robot} = 6.5$ ms and $T_{vs} = 200$ ms.

3.3.2 Validation

Ensuring that the motion prescribed by ANGUS' simulation was followed perfectly by the Cartesian robot in all situations proved unrealistic for several reasons. First, the dynamic model of ANGUS was highly nonlinear, therefore, its behaviour differed whether it was operated at low speeds (station-keeping case), or subject to sea current disturbances, or even at its highest accelerations. Second, the Cartesian robot's behaviour was also nonlinear and differed according to the demanded speeds on each axis. Finally, the (relatively) short range of the Cartesian robot's axes restricted the scope of motion which could be tested. As a result, the best performance which could be expected was that the combined dynamic system (Cartesian robot + ANGUS' simulation) responded in a manner *"closely resembling"* ANGUS' simulation. In particular, the demanded position and speed should match the actual position and speed.

Two types of experiments are now presented:

1. The first three tests assessed how the Cartesian robot followed the simulation when the latter received as thruster inputs the maximum power values, i.e. $\pm$ 100 %. The integration coefficients K_{ix} and K_{iy} were tuned so as to obtain the best possible results.

2. The second experiment checked whether the tuning so obtained still yielded a satisfactory behaviour during the visual servoing experiments of chapter 7.

For all experiments, the control coefficients were $K_{ix} = 1$ and $K_{iy} = 10$.

Test 1: surge dynamics

In the first test, the port and starboard thruster inputs were set to 100 % for 2.9 s (maximum forward acceleration) and then set to −100 % for 7.1 s (maximum backward acceleration). The objective was to provide the maximum accelerations on the X-axis of the Cartesian robot.

Figure 3.4 shows the desired X position of ANGUS (plain line) and the actual X position travelled by the Cartesian robot (pluses). Note that the order of magnitude of the small tracking error on the position is in the order of one centimetre.

Figure 3.5 plots the Y-axis, and does not exhibit any noticeable tracking error. In the test, the displacement on the Y-axis was very small (a few centimetres), and illustrated the coupling between the surge and sway degrees-of-freedom of ANGUS' dynamic model.

These two figures demonstrate clearly that the trajectory of ANGUS was followed.

The velocity profiles of ANGUS and the Cartesian robot are presented in figures 3.6 and 3.7 respectively. The "high" acceleration demanded on the X-axis could not be perfectly followed by the Cartesian robot. It was clear from the inspection of figure 3.6 that the response time of the X-axis was too slow, causing the Cartesian robot to overcompensate. It would reach a speed greater than that demanded at time $t = 4$ s. The same behaviour occurred again when the Cartesian robot slowed down before moving backwards. This figure clearly shows that it is not possible to replicate exactly the velocity profile of ANGUS' simulation for the surge degree-of-freedom at "high" accelerations. However, the control coefficient K_{ix} chosen did give the best results.

Even though the Y-axis velocity profile of ANGUS was perfectly followed, the demanded speed was only a few centimetres per second.

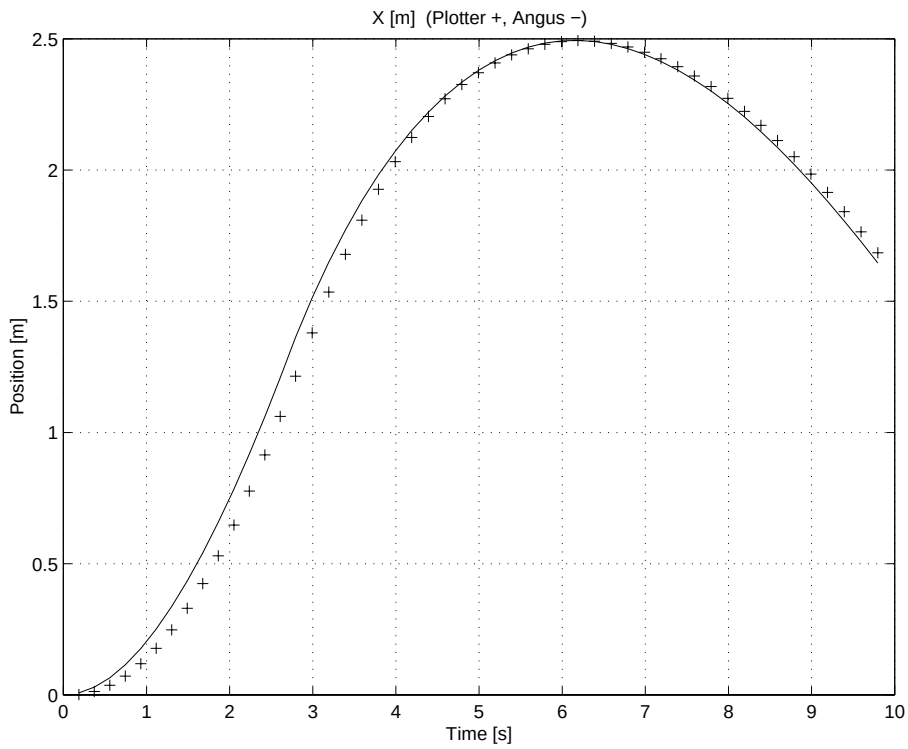


Figure 3.4: Full surge thrusters experiment: X displacement vs time.

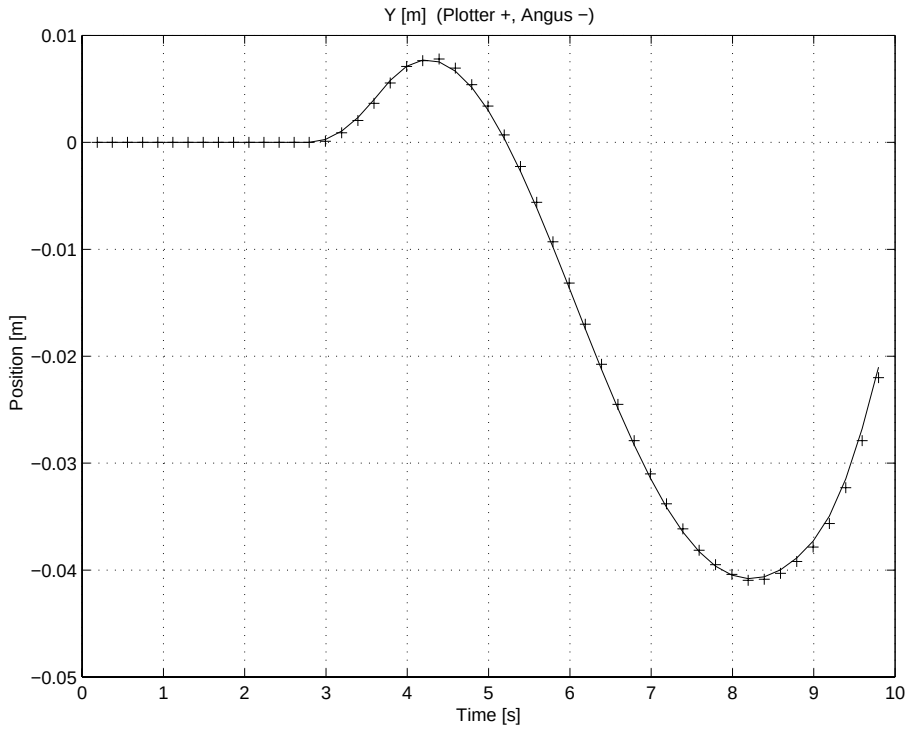


Figure 3.5: Full surge thrusters experiment: Y displacement vs time.

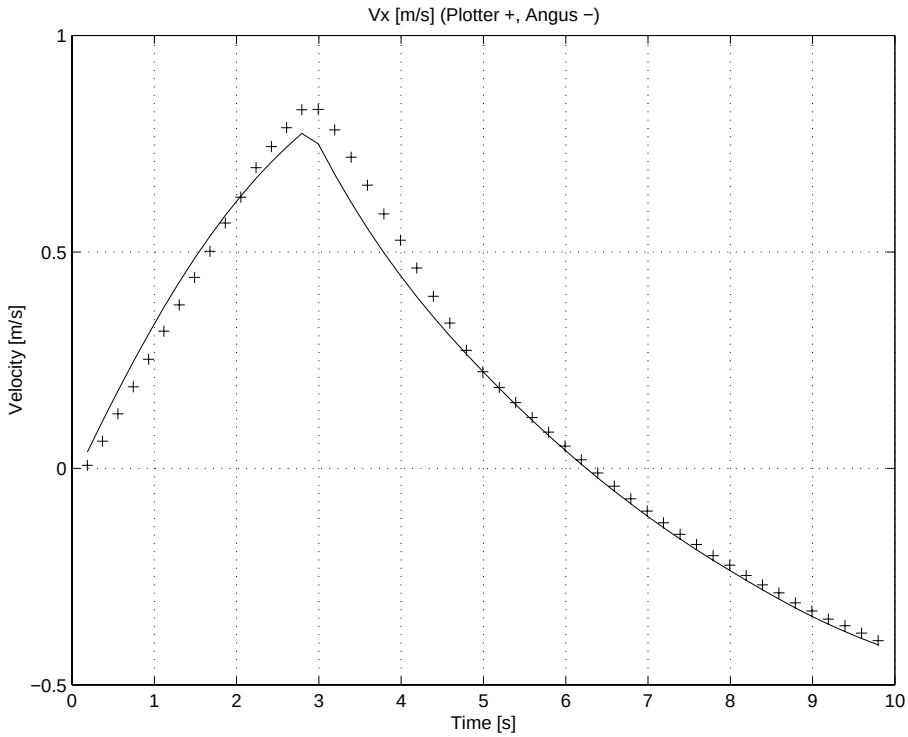


Figure 3.6: Full surge thrusters experiment: surge velocity vs time.

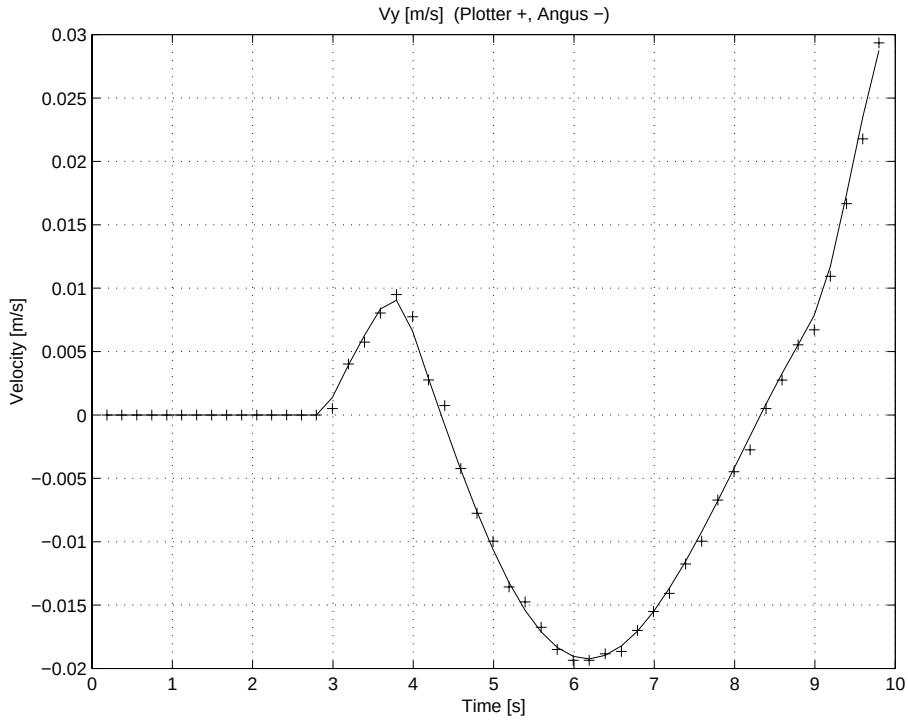


Figure 3.7: Full surge thrusters experiment: sway velocity vs time.

Test 2: sway dynamics

The second test assessed the Y-axis. The sway thruster was set to 100 % for 4.9 s and then reversed to −100 % for 5.1 s. As in the previous test, the desired position of ANGUS and the actual position of the Cartesian robot were recorded (figure 3.8 and 3.10), as well as the desired and actual velocities (figure 3.9 and 3.11).

A tracking error in both position and velocity (figures 3.8 and 3.9) is noticeable along the X-axis. This showed that the controller gains for the X-axis could not be tuned so that small velocities and high velocities yielded similar performances. However, in this particular case, the velocity and consequently the displacement were relatively small, lessening the negative impact of this result on ANGUS' emulation.

The dynamic response along the Y-axis of the Cartesian robot emulated well ANGUS' response. Figure 3.10 shows that the positioning error between ANGUS' model Y-position and the Cartesian robot Y-position was close to zero. A slight overshoot between the actual Cartesian robot's velocity and the demanded one can be observed at time $t = 5$ s, when the sway thruster changed sign (figure 3.11). The time response delay, which manifested itself as an overshoot, was due to the combined action of the axis inertia and of the mechanical slack on the axis. However, the high acceleration demand was well followed, especially after the thruster's sign change, with the exception of the first two measured points just after the overshoot.

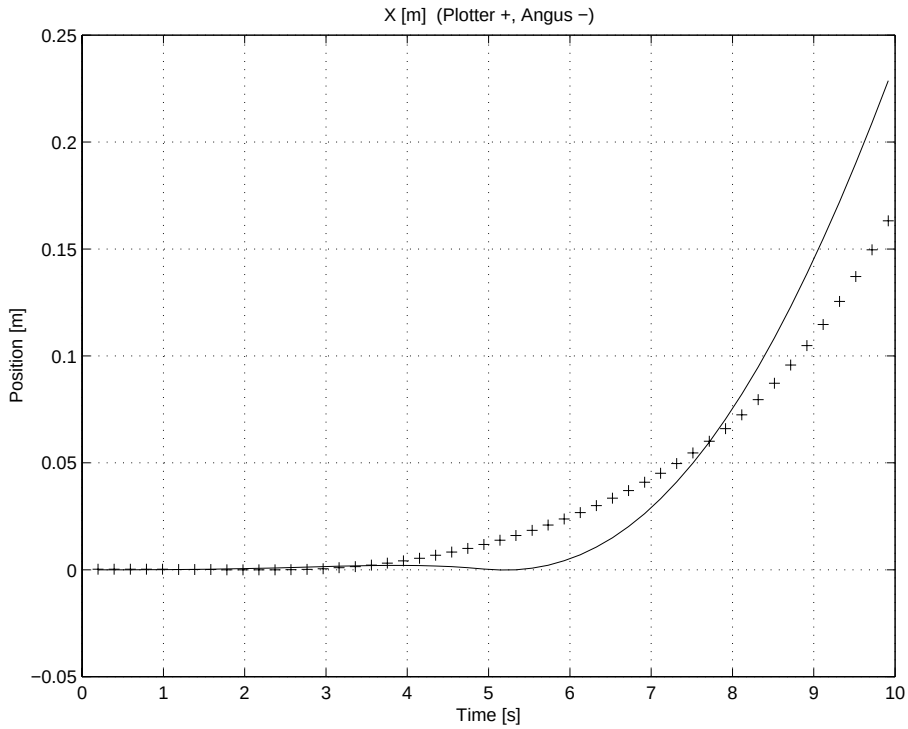


Figure 3.8: Full sway thrusters: X displacement vs time.

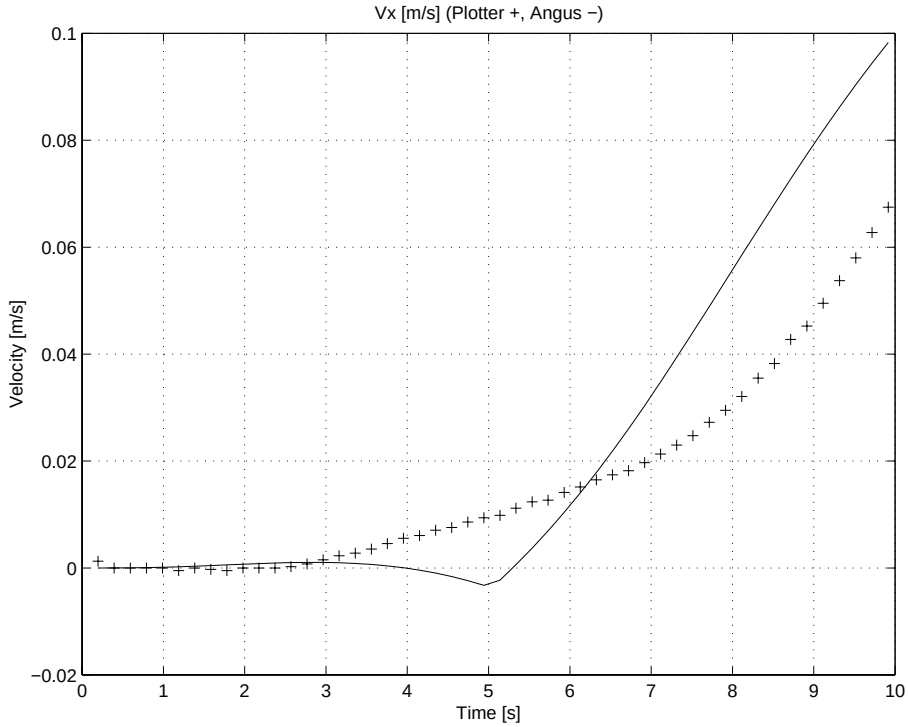


Figure 3.9: Full sway thrusters experiment: surge velocity vs time.

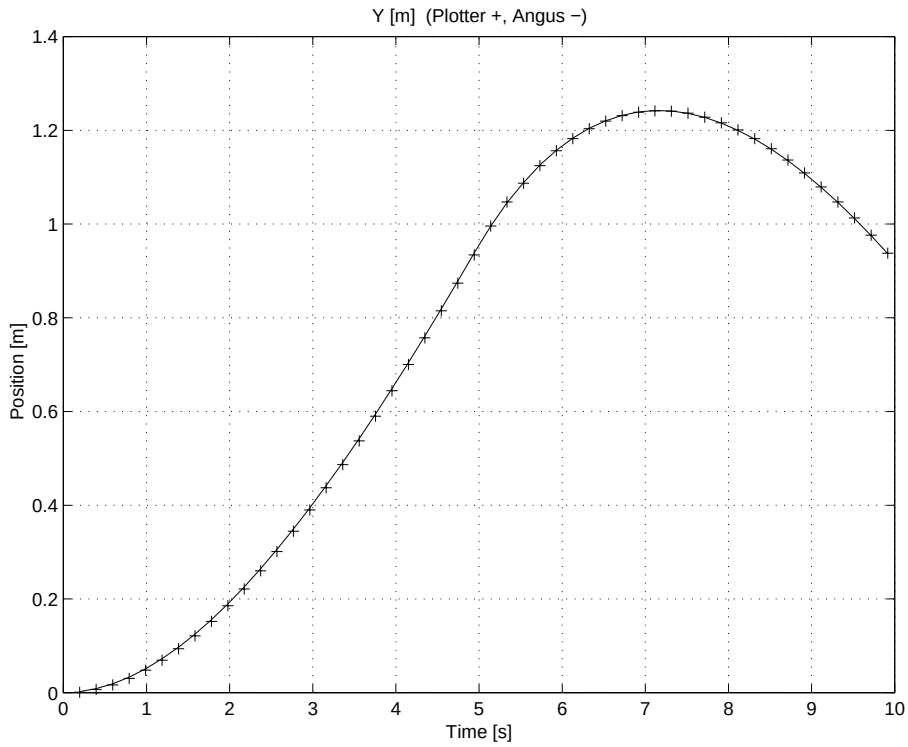


Figure 3.10: Full sway thrusters: Y displacement vs time.

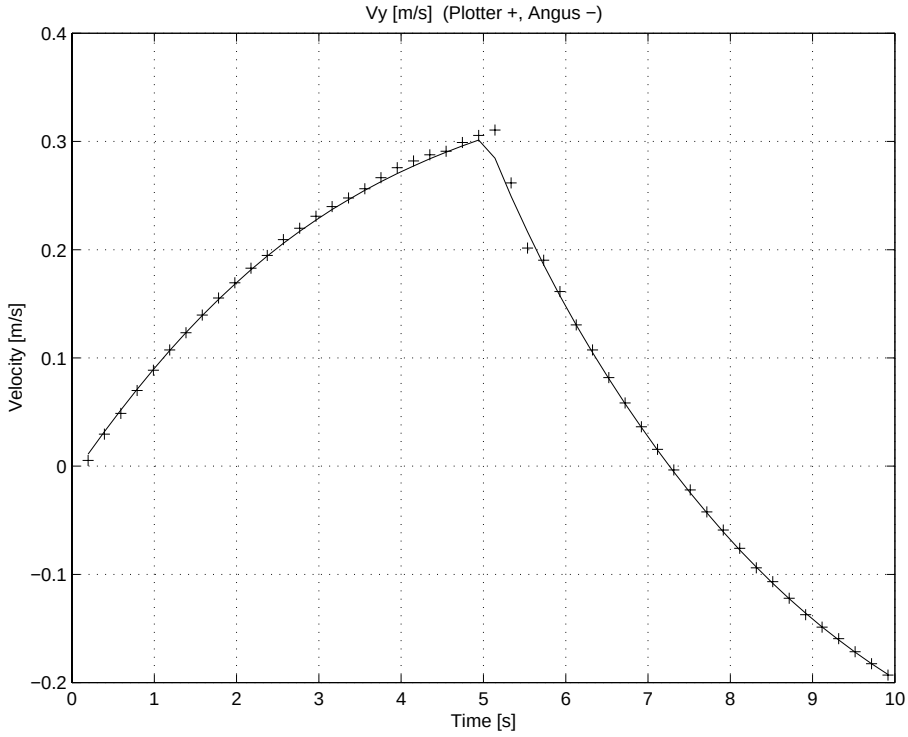


Figure 3.11: Full sway thrusters experiment: sway velocity vs time.

Test 3: combined surge and sway dynamics

This test illustrates the behaviour of the Cartesian robot when ANGUS' surge and sway thrusters were simultaneously used. In this third experiment, the port and starboard thrusters were concurrently set to 100 % for the first 3.9 s and then set to −100 % until the end. The sway thrusters were simultaneously set to 100 % during the first 4.9 s and then reversed to −100 %. Figures 3.12 and 3.13 illustrate the recorded positions of the ANGUS model and the Cartesian robot, figures 3.14 and 3.15 the recorded velocities.

On the X-axis, the same phenomenon as in test 2 can be observed: a tracking error and an overshoot of the Cartesian robot's X-velocity (figure 3.14). This velocity tracking error caused repercussions on the positioning error (figure 3.12). Nevertheless, the tracking error in the positioning remained within a few centimetres, which was acceptable.

Similarly, tracking errors of the Cartesian robot's Y-velocity (figure 3.15) within a few centimetres per second caused repercussions on the positioning error (figure 3.13). The latter also remained within a few centimetres. Again this is deemed acceptable for the purposes of this thesis.

These tests aimed at finding the best compromise of the PI controllers' gain values so that the Cartesian robot replicated ANGUS' simulation behaviour as *"closely"* as possible. These experiments showed that it was difficult to obtain good performances in both position and velocity. This thesis favours position over speed since the final objective of the visual servoing algorithms is to serve as a positioning function (station-keeping).

To conclude these tests, it is interesting to look at the behaviour of this emulation system in use during the actual visual servoing experiments of chapter 7. Notice how the behaviour is not affected by introducing vision in the loop.

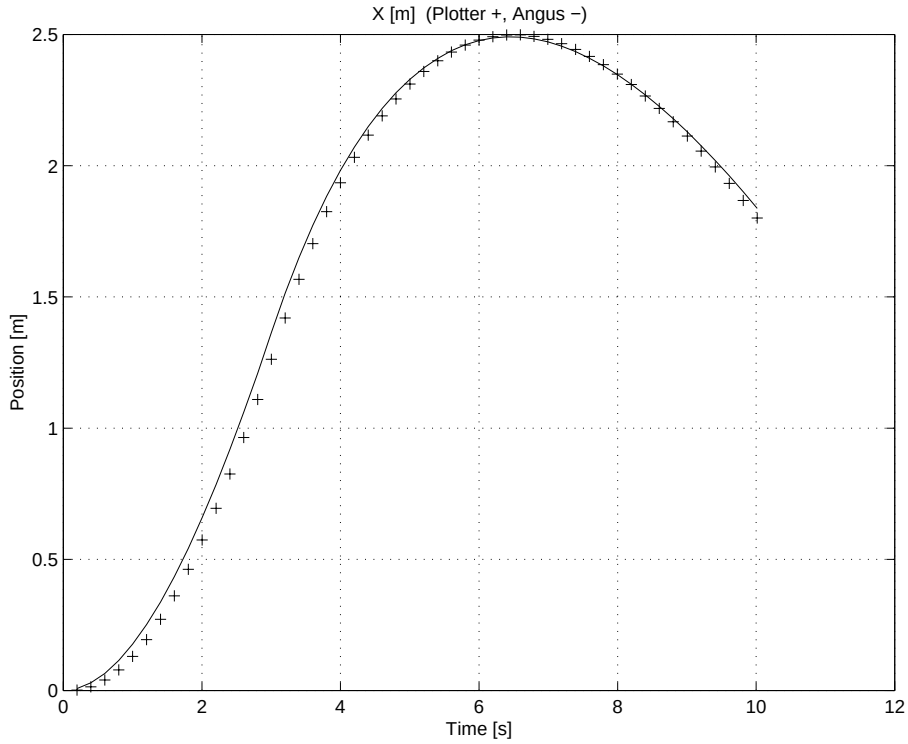


Figure 3.12: Combined surge and sway thrusters experiment: X displacement vs time.

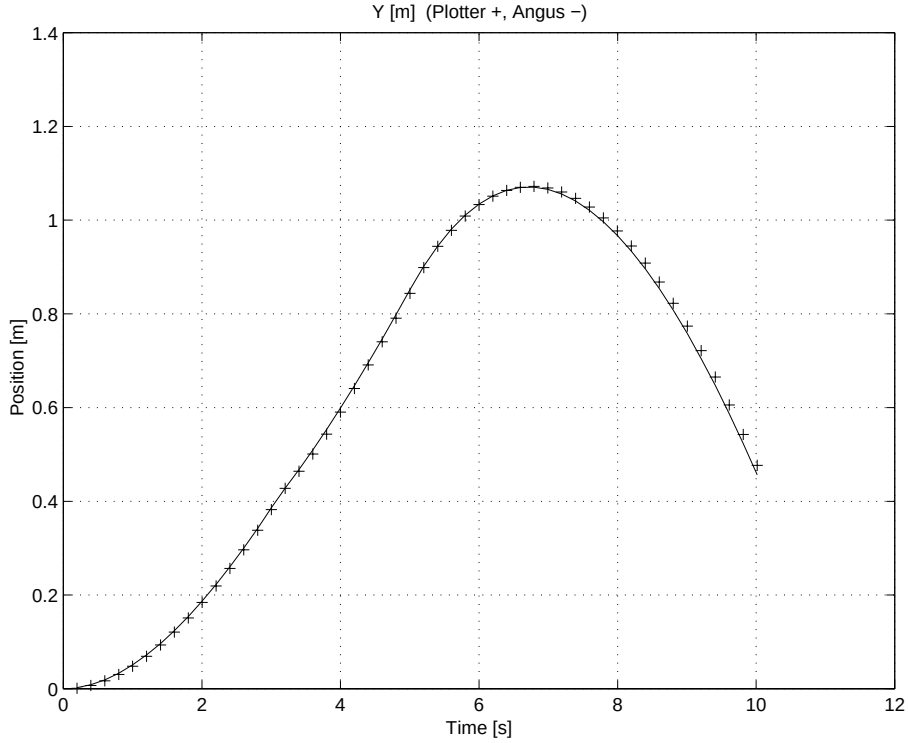


Figure 3.13: Combined surge and sway thrusters experiment: Y displacement vs time.

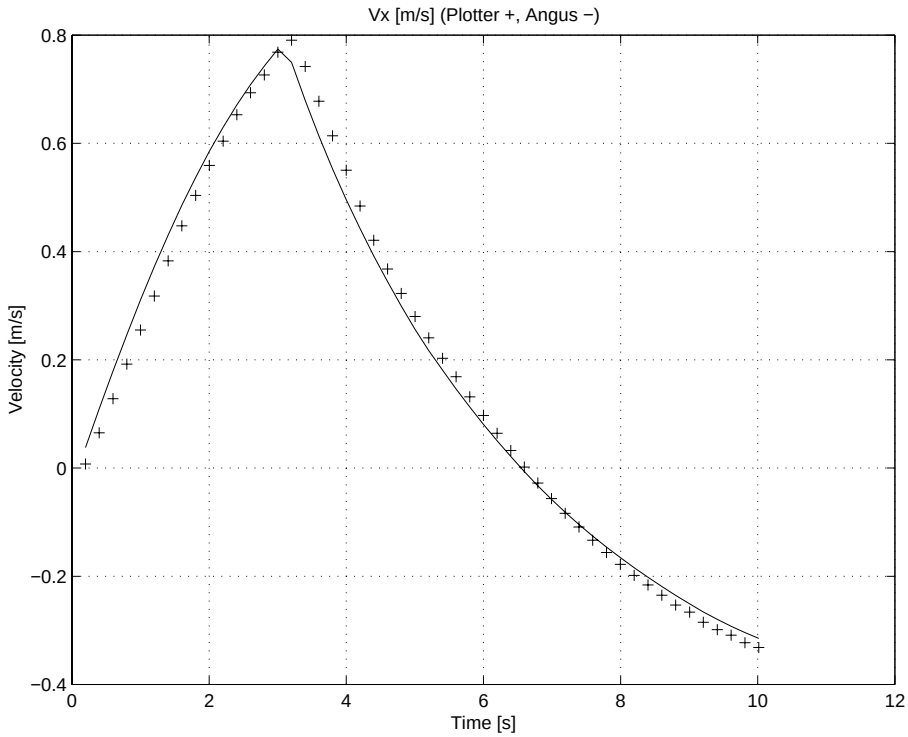


Figure 3.14: Combined surge and sway thrusters experiment: surge velocity vs time.

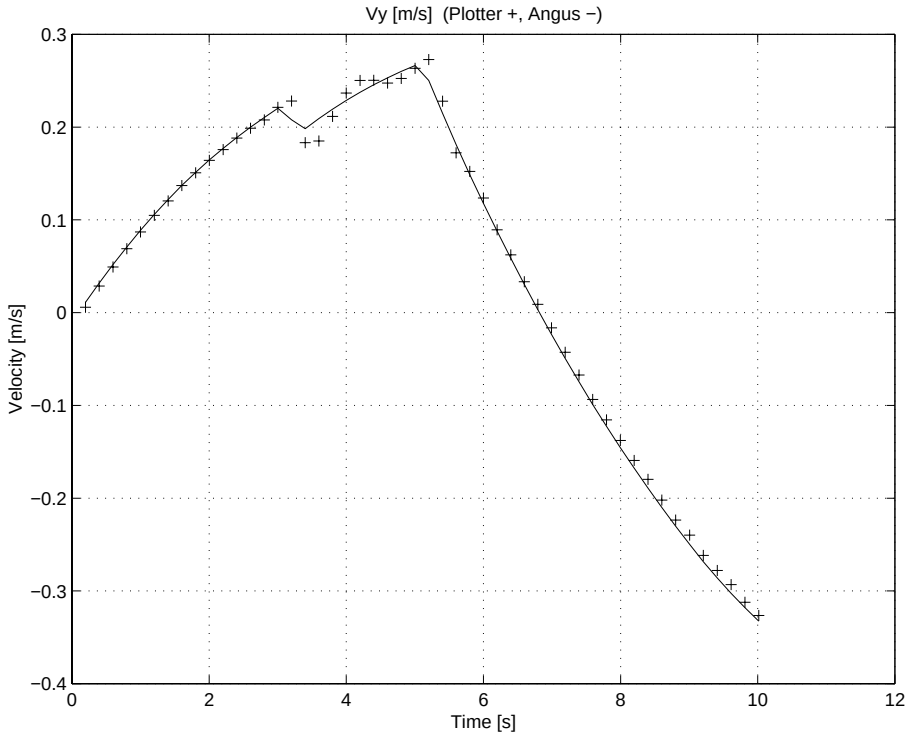


Figure 3.15: Combined surge and sway thrusters experiment: sway velocity vs time.

Experiment 2: visual servoing

In chapter 7 , this emulation method was used to conduct visual servoing experiments in real-time. The reader should refer to chapter 7 for details of the experimental setup. In this section, the comparison between the Cartesian robot's motion and the prescribed motion issued from ANGUS' simulation during one of the visual servoing experiment is made.

Again, as in the previous tests, it is clearly noticeable that the errors between the actual position and the demanded position were not significant (less than one centimetre). Figures 3.16 and 3.17 clearly demonstrate this claim. However, the errors between the actual speed and the prescribed speed were more significant (see figures 3.18 and 3.19). The Cartesian robot's speed signal was noisy. This was due to the fact that the visual servoing algorithm did not run on a real-time operating system and that the control loop sampling time T_{vs} was not constant. For example, on this experiment, the mean value of T_{vs} was 213 ms with a standard deviation of 115 ms!

The noise when evaluating the speed was increased, since the speed values were computed from the position measurements by a straight differentiation and divided by the sampling time.

The visual servoing test shows that the position of the UUV model was well replicated by the Cartesian robot, but the velocity profiles of the simulation and those of the Cartesian robot did not match.

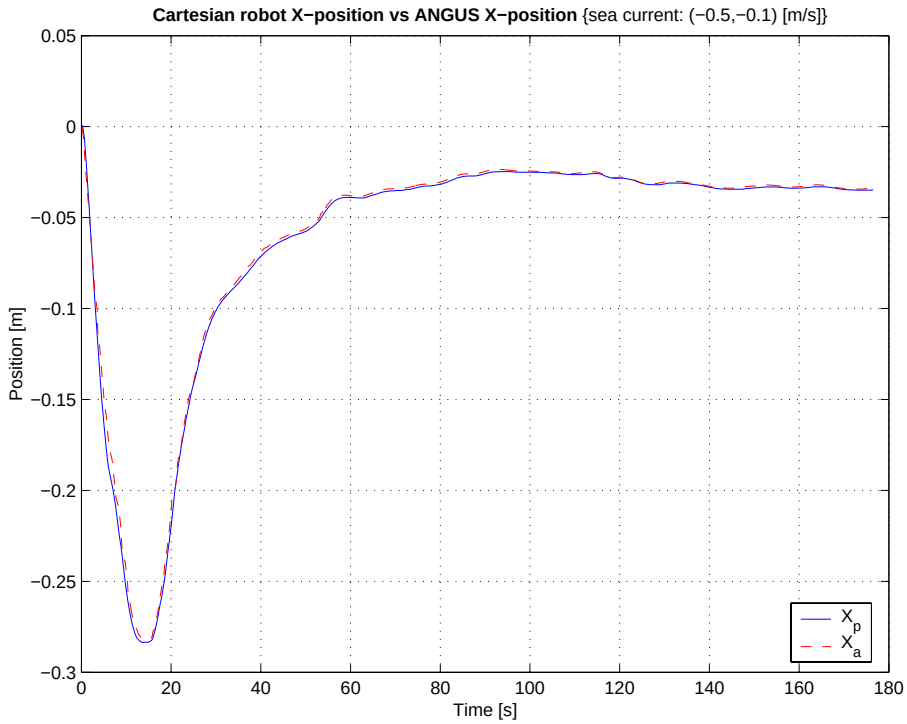


Figure 3.16: Comparison between the demanded X-position of ANGUS (red) and the actual X-position of the Cartesian robot (blue) vs time.

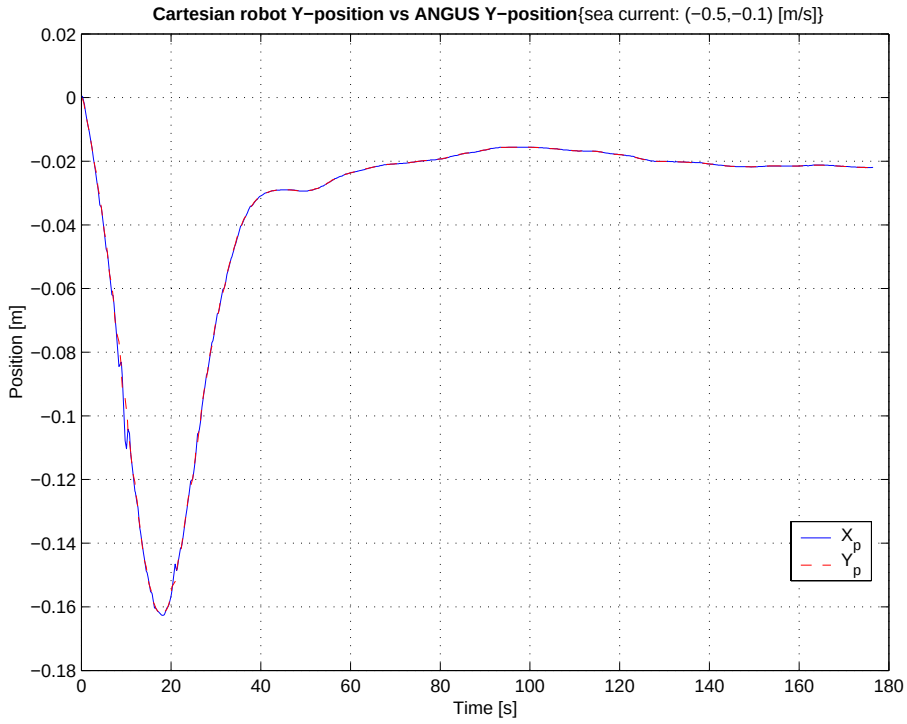


Figure 3.17: Comparison between the demanded Y-position of ANGUS (red) and the actual Y-position of the Cartesian robot (blue) vs time

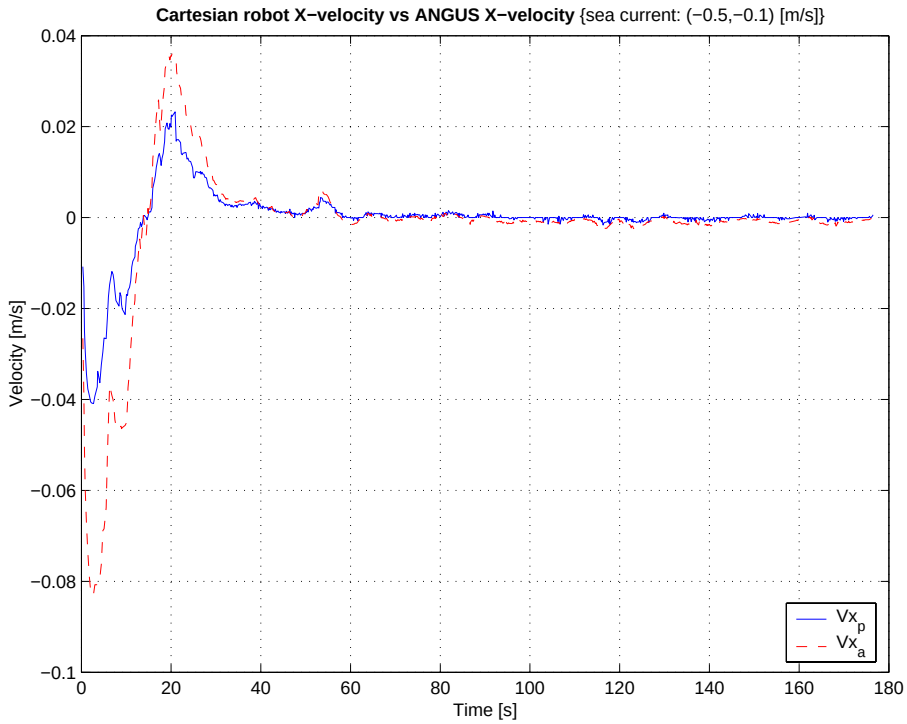


Figure 3.18: Comparison between the demanded X-velocity of ANGUS (red) and the actual X-velocity of the Cartesian robot (blue) vs time

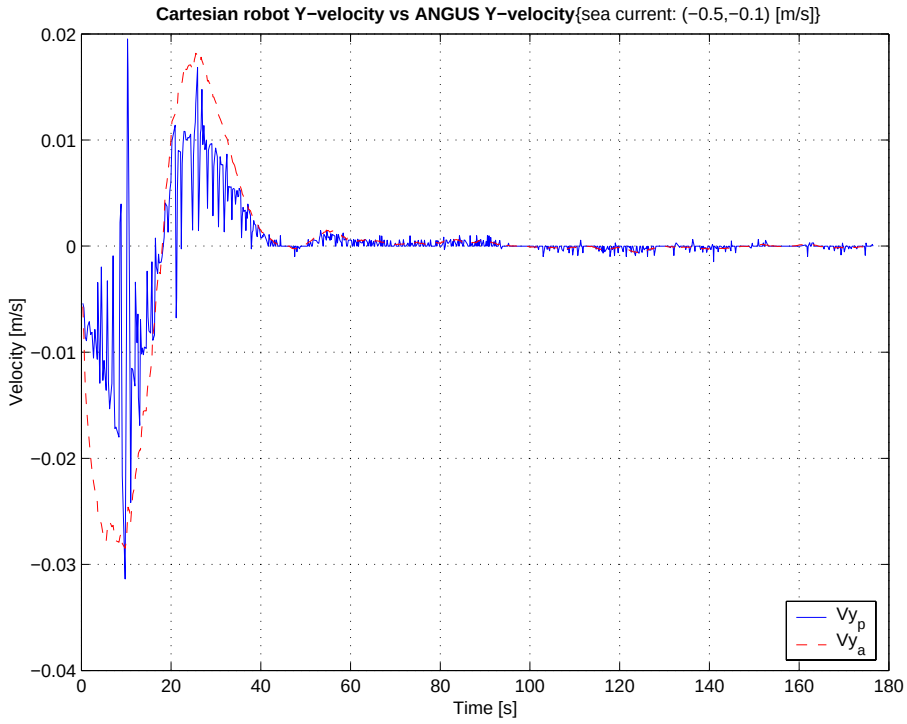


Figure 3.19: Comparison between the demanded Y-velocity of ANGUS (red) and the actual Y-velocity of the Cartesian robot (blue) vs time

3.4 Conclusions

The main contribution of this chapter was the emulation of the dynamic behaviour of an underwater vehicle on a Cartesian robot for real-time evaluation of underwater visual servoing techniques. The details of the real-time implementation were presented, as well as the experimental accuracy of such an emulation.

This chapter shows that the trajectory of the ANGUS simulation could be accurately followed, in particular during the visual servoing experiments described in chapter 7. However, the positioning accuracy had to be favoured over the velocity. In other words, by ensuring that both ANGUS and the Cartesian robot followed the same trajectory, the Cartesian robot was forced to be slightly slower than ANGUS. The complete system, including the ANGUS simulation and the Cartesian robot, could be considered as a realistic but slower version of the ANGUS UUV. Thus, the visual servoing experiments presented in chapter 7 illustrate the control of an hybrid system which behaved like a typical UUV, but not strictly like ANGUS, since the latter's speeds could not be followed exactly.

Chapter 4

Recovering motion from a pair of images with a calibrated camera

In this chapter, some background notions of projective geometry, and their application to the geometric modelling of a CCD camera are recalled. Since this thesis has focused on the control of an underwater robot equipped with a single camera, the geometry of a moving camera and how a set of corresponding feature points in two images describe the motion of a camera will be discussed.

4.1 Theoretical background

4.1.1 Co-ordinate transformations: the homogeneous matrices

In this section, the notion of the homogeneous matrices and their application to co-ordinate transformations between Euclidean frames is briefly introduced. **Note that only right-handed co-ordinate systems will be considered in this thesis.** Although left-handed co-ordinate systems are also common, especially within the computer graphics community. Let $\mathcal{F}^*$ be the Cartesian reference frame, which is also called the *desired* frame in the visual servoing framework. The aim is to express the co-ordinates of a 3-D point $\mathbf{M}^*$ in a new Cartesian frame $\mathcal{F}$. The geometrical relationship between $\mathcal{F}^*$ and $\mathcal{F}$ is illustrated in figure 4.1. If ${}^d\mathbf{R}_c$ is the rotation matrix rotating $\mathcal{F}^*$ onto $\mathcal{F}$, and ${}^d\mathbf{t}_c$ is the translation vector from $\mathcal{F}^*$ to $\mathcal{F}$, then the corresponding homogeneous transform matrix is:

$$ {}^d\mathbf{T}_c = \begin{bmatrix} {}^d\mathbf{R}_c & {}^d\mathbf{t}_c \\ 0 & 1 \end{bmatrix} \tag{4.1} $$

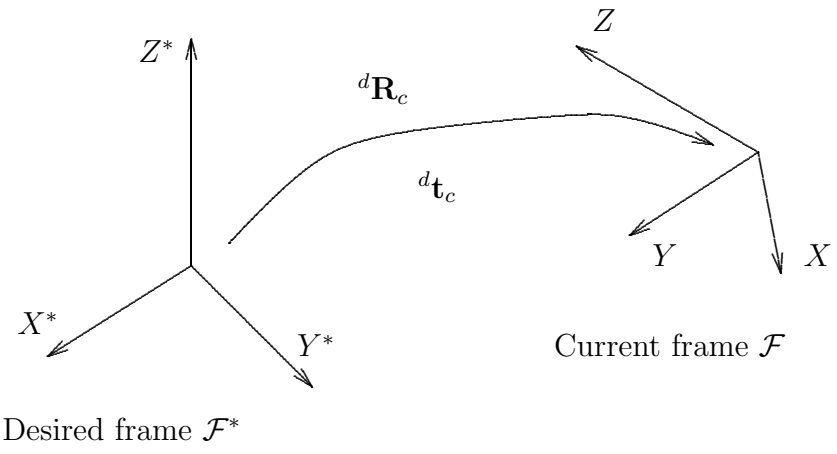


Figure 4.1: Co-ordinate frame transformation with homogeneous matrices.

With this notation, the co-ordinates $[x^*, y^*, z^*]^T$ of $\mathbf{M}^*$ in $\mathcal{F}^*$ are related to its co-ordinates $[x, y, z]^T$ in $\mathcal{F}$ by:

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} {}^d\mathbf{R}_c & {}^d\mathbf{t}_c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x^* \\ y^* \\ z^* \\ 1 \end{bmatrix} \tag{4.2}$$

The inverse homogeneous transformation ${}^d\mathbf{T}_c^{-1}$ is:

$${}^d\mathbf{T}_c^{-1} = \begin{bmatrix} {}^d\mathbf{R}_c^T & -{}^d\mathbf{R}_c^T {}^d\mathbf{t}_c \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} {}^c\mathbf{R}_d & {}^c\mathbf{t}_d \\ 0 & 1 \end{bmatrix} \tag{4.3}$$

4.1.2 Notation and projective geometry

In the following section, column vectors will be denoted by bold letters such as e.g. $\mathbf{t}$, the corresponding row vector will be its transpose: $\mathbf{t}^T$. A matrix will be denoted by upper case bold letters, e.g. $\mathbf{A}$, and a square identity matrix by $\mathbf{I}_n$, where n denotes the matrix dimension. A point in Euclidean space will be written like so: $\mathbf{M}$, its projective counterpart like this: $\tilde{\mathbf{M}}$.

Projective geometry is a mathematical framework commonly used to solve some geometric computer vision problems. This section provides a brief introduction to projective geometry. The reader is referred to the classical book of Faugeras [21] or

to the more recent one written by Zisserman and Hartley [32] for a more in-depth coverage of the subject, and to the references therein.

A point $\tilde{\mathbf{M}}$ in a projective space of dimension n, $\mathbb{P}^n$, is represented by a $(n+1)$-vector of co-ordinates $\mathbf{x} = [x_1, x_2, \ldots, x_{n+1}]^T$. By definition, two $(n+1)$-vectors $\mathbf{x}$ and $\mathbf{y}$ represent the same projective point $\tilde{\mathbf{M}}$, belonging to $\mathbb{P}^n$ if and only if $\exists\, \lambda \in \mathbb{R}$ so that $\mathbf{x} = \lambda\mathbf{y}$. For this reason, the co-ordinates of a projective point are called *homogeneous co-ordinates.*

A projective space $\mathbb{P}^n$ can be seen as an extension of an n-dimensional Euclidean space $\mathbb{R}^n$. Indeed, an Euclidean point $\mathbf{M} = [x_1, \ldots, x_n]^T \in \mathbb{R}^n$ can be described in the projective space $\mathbb{P}^n$ by a point $\tilde{\mathbf{M}}$ using the homogeneous co-ordinates $[x_1, \ldots, x_n, 1]^T \in \mathbb{R}^{n+1}$. In other words, any projective point with a non-zero $(n+1)$th co-ordinate represents an Euclidean point. Conversely, given a projective point $\tilde{\mathbf{M}} = [x_1, \ldots, x_{n+1}]^T$ with $x_{n+1} \neq 0$, its corresponding Euclidean point is $\mathbf{M} = [\frac{x_1}{x_{n+1}}, \ldots, \frac{x_n}{x_{n+1}}]^T$. Projective points with $x_{n+1} = 0$ are called *points at infinity* and can be considered as directions of lines in $\mathbb{R}^n$.

By definition, a projective transformation from a space $\mathbb{P}^n$ to a space $\mathbb{P}^m$ is represented by a $(m+1) \times (n+1)$ matrix. Since the co-ordinates of projective points are homogeneous, this matrix is unique up to a scale factor. Any non-singular $n \times n$ matrix represents an invertible linear projective transformation of $\mathbb{P}^n$ on itself, called *homography* or *collineation*. The particular case of homographies in the projective plane $\mathbb{P}^2$ shall be discussed in more details in section 4.3.1.

In a projective space $\mathbb{P}^n$, points and $(n-1)$-dimensional subspaces, called *hyperplanes*, are *dual*. Indeed, a $(n+1)$-dimensional vector can represent either a point or a hyperplane, and properties holding for a point will also hold for a hyperplane, and vice versa. In the case of the projective plane $\mathbb{P}^2$, lines and points are dual entities.

Also in $\mathbb{P}^2$, a line $\mathbf{l}$ of equation $ax+by+cz = 0$ is represented by the vector $[a, b, c]^T$. A point $\tilde{\mathbf{m}} = [x, y, z]^T$ belongs to $\mathbf{l}$ if and only if $\mathbf{l}^T\tilde{\mathbf{m}} = 0$. In addition, given two projective points $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{q}}$, the line passing through them is $\tilde{\mathbf{m}} \times \tilde{\mathbf{q}}$.

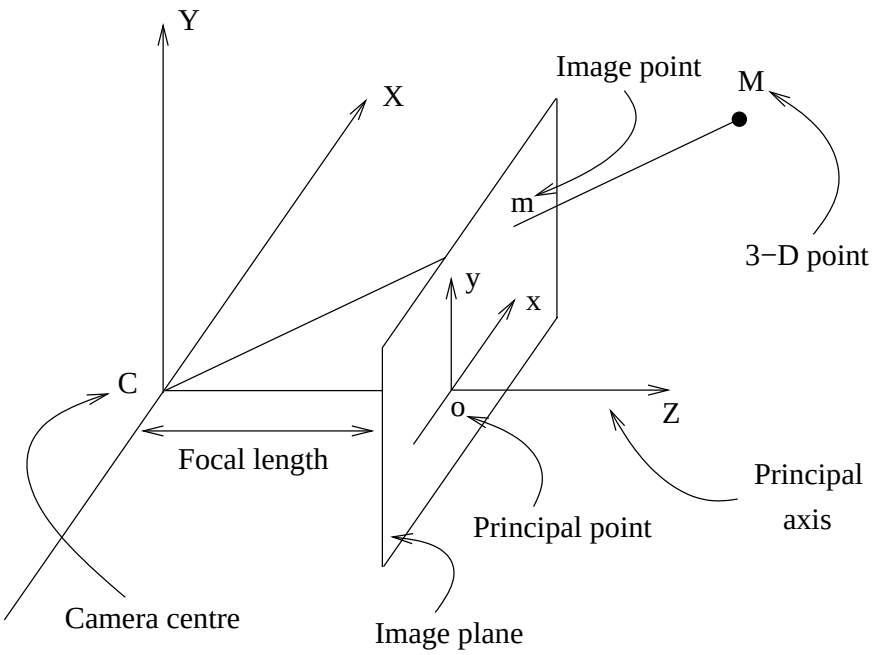


Figure 4.2: Projective camera model.

4.1.3 Camera modelling

In this section, the classical pinhole camera model, widely used in the computer vision literature for CCD cameras is discussed. It is shown that it can be represented as a linear projective mapping from the projective space $\mathbb{P}^3$ to the projective plane $\mathbb{P}^2$. This modelling only accounts for the geometry of the imaging process. It does not take into account the radiometric aspects of the scene. For an introduction on the latter, the reader is referred to [84] and the references therein.

The *pinhole camera model* is a central projection of points in space onto a plane. Let the centre of projection $\mathbf{C}$ be the origin of an Euclidean co-ordinate system, and consider the plane of equation $z = f$, which is called the *image plane* or *focal plane* (see figure 4.2). Under the pinhole camera model, a 3-D Euclidean point $\mathbf{M} = [X, Y, Z]^T$ of the scene is mapped to the 2-D Euclidean point $\mathbf{m}$ on the focal plane. The intersection of the line joining the points $\mathbf{C}$ and $\mathbf{M}$ and the plane of equation $z = f$ defines $\mathbf{m}$. The co-ordinates of $\mathbf{m}$ expressed in the focal plane reference frame $(\mathbf{o}xy)$ are therefore $[f\frac{X}{Z}, f\frac{Y}{Z}]^T$.

The centre of projection $\mathbf{C}$ is called *camera centre* or *optical centre*. The Z-axis, perpendicular to the image plane, is called the *principal axis* or *optical axis* of the

camera, and the point **o** where the principal axis meets the image plane is called the *principal point*. The plane passing through the camera centre and parallel to the image plane is called the *principal plane*.

If the scene point **M** and its projection onto the image plane are represented by homogeneous vectors (see 4.1.2), the mapping from the scene to the image becomes linear and can be expressed by a 3×4 matrix thus:

$$\tilde{\mathbf{m}} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \tilde{\mathbf{P}}_o \tilde{\mathbf{M}} \tag{4.4}$$

where $\tilde{\mathbf{M}} = [X, Y, Z, 1]^T$ and $\tilde{\mathbf{m}} = [f\frac{X}{Z}, f\frac{Y}{Z}, w]^T$ $(w \neq 0)$ are the homogeneous counterparts of the aforementioned Euclidean points **M** and **m** respectively. The 3×4 matrix $\tilde{\mathbf{P}}_o$ is called the *canonical projective matrix* and reads:

$$\tilde{\mathbf{P}}_o = [\mathbf{I}_3 \,|\, \mathbf{0}] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{4.5}$$

Expression (4.5) assumes that the origin of the co-ordinates in the image plane is at the principal point. This is not the case in general, and consequently an offset must be included. This is done simply in homogeneous co-ordinates. If $\mathbf{o} = [p_x, p_y, f]^T$ is the principal point, then the central projection equations read:

$$\tilde{\mathbf{m}} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} \tilde{\mathbf{P}}_o \tilde{\mathbf{M}}. \tag{4.6}$$

This model assumes that the image co-ordinates are Euclidean co-ordinates having equal scales in both axial directions. However, this assumption is not held for CCD cameras, and an additional scaling must be added. In particular, if the number of pixels per unit distance in image co-ordinates are k_x and k_y in the x and y directions, then eq. (4.6) becomes:

$$\tilde{\mathbf{m}} = \begin{bmatrix} \alpha_x & 0 & x_o \\ 0 & \alpha_y & y_o \\ 0 & 0 & 1 \end{bmatrix} \tilde{\mathbf{P}}_o \tilde{\mathbf{M}} \tag{4.7}$$

where $\alpha_x = fk_x$ and $\alpha_y = fk_y$ represent the *focal length in pixels* in the x and y directions respectively. Similarly, $\mathbf{o} = [x_o, y_o]^T$ is the *principal point in pixels*, with co-ordinates $x_o = k_x p_x$ and $y_o = k_y p_y$.

If the x and y axis of the camera are not perpendicular, an added parameter s called *skew* is incorporated in the projection equations:

$$\tilde{\mathbf{m}} = \begin{bmatrix} \alpha_x & s & x_o \\ 0 & \alpha_y & y_o \\ 0 & 0 & 1 \end{bmatrix} \tilde{\mathbf{P}}_o \tilde{\mathbf{M}} = \mathbf{A} \tilde{\mathbf{P}}_o \tilde{\mathbf{M}}. \tag{4.8}$$

The matrix $\mathbf{A}$ is called the *camera calibration matrix*. The parameters of $\mathbf{A}$ are called *intrinsic parameters* of the camera.

In this equation, the world reference frame and the camera reference frame (also called *standard camera frame*) coincide. This is not the case in general. If the rigid transformation from the world reference frame to the camera frame is defined by the rotation matrix ${}^w\mathbf{R}_c$, and the translation vector ${}^w\mathbf{t}_c$, then the projective equations read:

$$\tilde{\mathbf{m}} = \mathbf{A} \left[{}^w\mathbf{R}_c | - {}^w\mathbf{R}_c {}^w\mathbf{t}_c \right] \tilde{\mathbf{M}} = \tilde{\mathbf{P}} \tilde{\mathbf{M}} \tag{4.9}$$

where the 3×4 matrix $\tilde{\mathbf{P}}$ is termed the *camera projection matrix* and encodes all the information needed to project a world point of the scene onto its image in the focal plane of the camera. The parameters of ${}^w\mathbf{R}_c$ and ${}^w\mathbf{t}_c$ which relate the camera orientation and position (its *pose*) with respect to the world (scene) co-ordinate system are called the *extrinsic parameters*.

It is possible to retrieve the intrinsic and extrinsic parameters from the camera projection matrix $\tilde{\mathbf{P}}$, as well as the equations of the image plane, the principal axis, and the principal point. The derivation of these expressions falls beyond the scope of this thesis; the reader is therefore referred to [32].

In this section, the geometric modelling of a perspective camera based on the pin-hole model has been described. This allows the understanding of the relationship of scene points with their images on the CCD array. The following sections will describe the geometric relationships between image points in two views taken by a camera undergoing a rigid displacement. In section 4.2, a general 3-D scene shall be considered, and in section 4.3 the special case of planar scenes shall be described.

4.2 Epipolar geometry

The epipolar geometry is the intrinsic geometry between two views, whether they are taken by two different cameras (the stereo case) or by a single moving camera. The epipolar geometry is independent of scene structure, and only depends on the camera's intrinsic and extrinsic parameters.

In the following sections the epipolar geometry is first described and the terminology associated to it is introduced. The fundamental equation relating corresponding points in two views and its key algebraic element, also described with these tools, the fundamental matrix, is derived. Finally, a classical method to estimate the fundamental matrix is detailed.

4.2.1 Description

A problem arising in stereo vision is the search for point correspondences in two views: the *stereo matching problem*. These correspondences are related by the epipolar geometry. Assume that a point $\mathbf{M}$ in Euclidean 3-space is imaged as $\mathbf{m}$ in the first view and $\mathbf{m}'$ in the second view, as depicted in figure 4.3. The points $\mathbf{M}$, $\mathbf{m}$, $\mathbf{m}'$, $\mathbf{C}$, and $\mathbf{C}'$ are coplanar since $\mathbf{m}$ (resp. $\mathbf{m}'$) is the intersection of the optical ray $\mathbf{CM}$ (resp. $\mathbf{C}'\mathbf{M}$), and $\mathbf{M}$ is common. This plane is called $\mathbf{\Pi}$. Assume that $\mathbf{m}$ only is known. The aim is to try to determine where its corresponding point $\mathbf{m}'$ lies in the second view. The plane $\mathbf{\Pi}$ is defined by the baseline ($\mathbf{CC}'$) and the ray passing through $\mathbf{m}$. It is known that the ray corresponding to the unknown image point $\mathbf{m}'$ lies in $\mathbf{\Pi}$, hence the point $\mathbf{m}'$ lies on the line of intersection $\mathbf{l}'$ of $\mathbf{\Pi}$ with the second

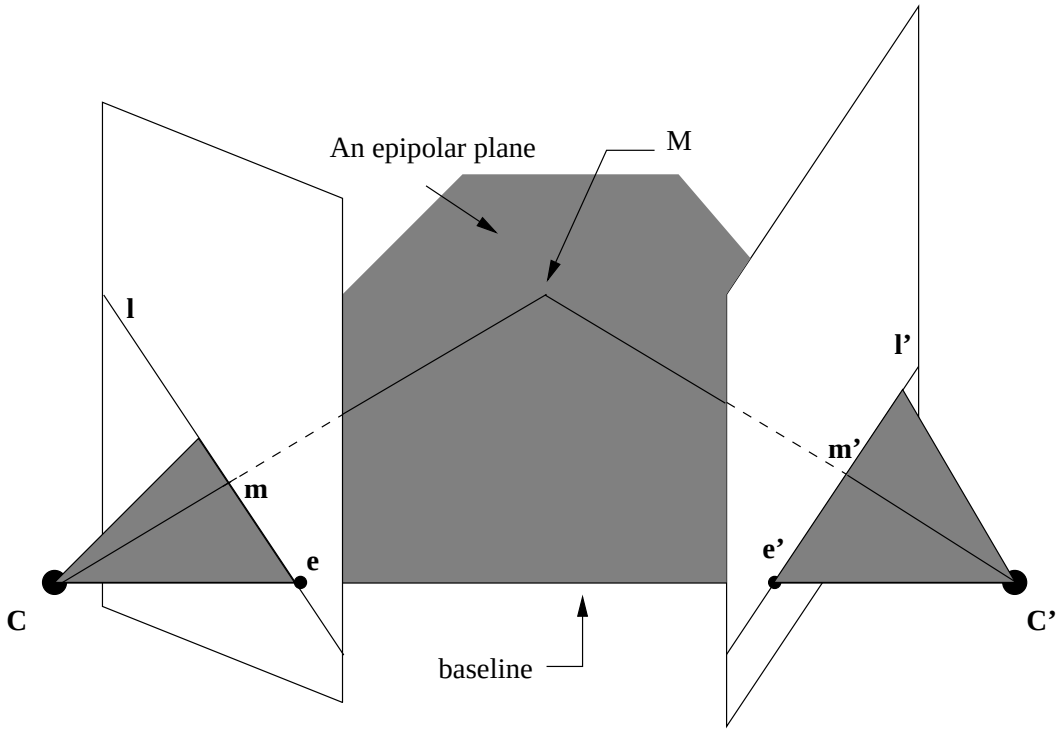


Figure 4.3: **The epipolar geometry:** The camera baseline intersects each image plane at the epipole $\mathbf{e}$ and $\mathbf{e}'$. Any plane $\mathbf{\Pi}$ containing the baseline is an epipolar plane. The epipolar lines $\mathbf{l}$ and $\mathbf{l}'$ are the intersection of the epipolar plane with the image planes. $\mathbf{m}$ and $\mathbf{m}'$ are the projections of 3-D point $\mathbf{M}$ on the image planes.

image plane. This line $\mathbf{l}'$ is the image in the second view of the ray back-projected from $\mathbf{m}$. This means that the search for the corresponding point to $\mathbf{m}$ is constrained to a line: $\mathbf{l}'$, which greatly reduces the search space. Any such line is termed *epipolar line*. $\mathbf{l}'$ is an epipolar line of the second camera. All epipolar lines in each image intersect at one special point called the *epipole*, as any epipolar plane must contain the baseline.

To summarise, a few definitions of the geometric entities associated with the epipolar geometry are given here and illustrated in figure 4.3:

- The *epipole* is the point of intersection of the line joining the camera centres (the baseline) with the image plane. It is also the image in one view of the camera centre of the other view.
- An *epipolar plane* is a plane containing the baseline and the 3-D point being projected.
- An *epipolar line* is the intersection of an epipolar plane with an image plane.

All epipolar lines intersect at the epipole. An epipolar plane intersects the left and the right image planes in epipolar lines, and defines the correspondence between the lines.

The next section will show that the fundamental matrix $\mathbf{F}$ is a 3×3 matrix of rank 2 which encodes the epipolar geometry.

4.2.2 The fundamental matrix

The ray back-projected from $\tilde{\mathbf{m}}$ by the camera matrix $\tilde{\mathbf{P}}$ is obtained by solving the equation $\tilde{\mathbf{P}}\,\tilde{\mathbf{M}} = \tilde{\mathbf{m}}$. The set of solutions is parameterised by a scalar λ so that:

$$\tilde{\mathbf{M}}(\lambda) = \tilde{\mathbf{P}}^{+}\,\tilde{\mathbf{m}} + \lambda\,\tilde{\mathbf{C}} \tag{4.10}$$

where $\tilde{\mathbf{P}}^{+} = \tilde{\mathbf{P}}^{T}(\tilde{\mathbf{P}}\tilde{\mathbf{P}}^{T})^{-1}$ is the pseudo inverse of $\tilde{\mathbf{P}}$, and $\tilde{\mathbf{C}}$ its null vector (i.e. $\tilde{\mathbf{P}}\,\tilde{\mathbf{C}} = 0$): the camera centre. The camera centre $\tilde{\mathbf{C}}$ and the point $\tilde{\mathbf{P}}^{+}\,\tilde{\mathbf{m}}$ (for $\lambda = 0$) lie on this ray. Now, these two points are projected by the second camera $\tilde{\mathbf{P}}'$ onto the epipolar line $\mathbf{l}'$ at $\tilde{\mathbf{P}}'\,\tilde{\mathbf{C}}$ and $\tilde{\mathbf{P}}'\,\tilde{\mathbf{P}}^{+}\,\tilde{\mathbf{m}}$. As seen in section 4.1.2, the equation of $\mathbf{l}'$ is obtained with the cross product of these two points, namely $\mathbf{l}' = (\tilde{\mathbf{P}}'\,\tilde{\mathbf{C}}) \times (\tilde{\mathbf{P}}'\,\tilde{\mathbf{P}}^{+})\,\tilde{\mathbf{m}}$. By definition, the point $\tilde{\mathbf{P}}'\,\tilde{\mathbf{C}}$ is the epipole in the second image which will be denoted as $\tilde{\mathbf{e}}'$. Consequently, $\mathbf{l}' = [\tilde{\mathbf{e}}']_{\times}\,\tilde{\mathbf{P}}'\,\tilde{\mathbf{P}}^{+}\,\tilde{\mathbf{m}} = \mathbf{F}\,\tilde{\mathbf{m}}$, where $\mathbf{F}$ is the matrix:

$$\mathbf{F} = [\tilde{\mathbf{e}}']_{\times}\,\tilde{\mathbf{P}}'\,\tilde{\mathbf{P}}^{+} \tag{4.11}$$

Eq. (4.11) defines the *fundamental matrix* $\mathbf{F}$. If the two camera centres $\tilde{\mathbf{C}}$ and $\tilde{\mathbf{C}}'$ coincide, then $\tilde{\mathbf{P}}'\,\tilde{\mathbf{C}}'$ and the fundamental matrix are null. This corresponds to a pure rotation about the optical centre of the camera. This algebraic derivation of the fundamental matrix was found in [93]. A geometric derivation can be found in [32].

The fundamental matrix has a few basic properties that will be briefly recalled. The most important being the *correspondence condition*.

Correspondence condition: *for any pair of corresponding points* $\tilde{\mathbf{m}} \leftrightarrow \tilde{\mathbf{m}}'$ *in two images*,

$$\tilde{\mathbf{m}}'^T \mathbf{F}\, \tilde{\mathbf{m}} = 0. \tag{4.12}$$

This condition is important since it shows that the fundamental matrix can be estimated from image point correspondences alone (see 4.2.3).

To summarise the properties of the fundamental matrix:

- $\mathbf{F}$ is a rank 2 homogeneous matrix with 7 degrees-of-freedom.
- **Point correspondence:** if $\tilde{\mathbf{m}}$ and $\tilde{\mathbf{m}}'$ are corresponding image points, then $\tilde{\mathbf{m}}'^T \mathbf{F}\, \tilde{\mathbf{m}} = 0$.
- **Epipolar lines:**
  - $\tilde{\mathbf{l}}' = \mathbf{F}\, \tilde{\mathbf{m}}$ is the epipolar line corresponding to $\tilde{\mathbf{m}}$.
  - $\tilde{\mathbf{l}} = \mathbf{F}^T \tilde{\mathbf{m}}'$ is the epipolar line corresponding to $\tilde{\mathbf{m}}'$.
- **Epipoles:**
  - $\mathbf{F}\, \tilde{\mathbf{e}} = 0$.
  - $\mathbf{F}^T \tilde{\mathbf{e}}' = 0$.

The $\mathbf{F}$ matrix can be computed using a classical linear technique [45]. This technique shall now be presented.

4.2.3 The normalised 8-point algorithm

Eq. (4.12) suggests that $\mathbf{F}$ can be determined from point correspondences alone. This is done by using appropriate normalisation [33] in conjunction with the algorithm described in [45]. Nonlinear estimation algorithms which would require iterative methods have not been considered because of their computational cost. Nevertheless pointers to the appropriate literature on the subject will be given in section 4.4.

The starting point of the 8-point algorithm is eq. (4.12) which holds true for any pair of matching points in two images. If $\tilde{\mathbf{m}} = [x, y, 1]^T$ and $\tilde{\mathbf{m}}' = [x', y', 1]^T$ are

corresponding points, eq. (4.12) gives rise to one linear equation in the unknown entries of $\mathbf{F}$. At least seven point matches are required to solve for $\mathbf{F}$. Each pair of corresponding points yields the following equation:

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0. \quad (4.13)$$

If $\mathbf{f}$ denotes the 9-vector made up of the entries of the fundamental matrix so that $\mathbf{f} = [f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33}]^T$, then for a set of n points matches, the following linear system must be solved:

$$\mathbf{Q}\,\mathbf{f} = \begin{bmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{bmatrix} \mathbf{f} = 0 \quad (4.14)$$

For a solution to this homogeneous system, the system matrix $\mathbf{Q}$ must have at most rank 8 (otherwise, $\mathbf{f} = 0$ is the only solution!). If the data are exact (no noise) and the rank of $\mathbf{Q}$ is exactly 8, then the solution is unique (up to some scale). However, point matches will be noisy and the system matrix may have rank 9. In this case, a least-squares solution for $\mathbf{f}$ must be found. It is well known that the singular vector corresponding to the smallest singular value of $\mathbf{Q}$ is the solution to eq. (4.14) which minimises $||\mathbf{Q}\,\mathbf{f}||$ subject to a constraint such as $||\mathbf{f}|| = 1$. Other constraints are possible [32]. If the singular value decomposition of $\mathbf{Q}$ is $\mathbf{Q} = \mathbf{U}\,\mathbf{D}\,\mathbf{V}^T$, then the least-squares solution $\mathbf{f}$ of the system (4.14) corresponds to the last column of the matrix $\mathbf{V}$.

In general, the system matrix $\mathbf{Q}$ is formed by pixel points, and is consequently ill-conditioned. Hartley [33] proposed a simple normalisation which greatly improved the conditioning of the system. It consists of a translation and a scaling: the data (for each image) are centred in their barycentrum, and scaled so that the RMS distance of the points to the origin is $\sqrt{2}$ (*isotropic scaling*).

Deriving the fundamental matrix in this way does not guarantee the property that its rank is 2 as stated in section 4.2.2. A matrix $\mathbf{F}'$ which obeys that constraint is one which minimises the Frobenius norm $||\mathbf{F}' - \mathbf{F}||$, and is subject to $\det(\mathbf{F}') = 0$.

The normalised 8-point algorithm is summarised below.

1. Transform the image co-ordinates according to:

$$\begin{aligned} \tilde{\mathbf{m}}_n &= \mathbf{T}\,\tilde{\mathbf{m}} \\ \tilde{\mathbf{m}}'_n &= \mathbf{T}'\,\tilde{\mathbf{m}}' \end{aligned}$$

where the transformation matrices $\mathbf{T}$ and $\mathbf{T}'$ have the following expression:

$$\mathbf{T} = \begin{bmatrix} 1/d & 0 & -c_x/d \\ 0 & 1/d & -c_y/d \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{T}' = \begin{bmatrix} 1/d' & 0 & -c'_x/d' \\ 0 & 1/d' & -c'_y/d' \\ 0 & 0 & 1 \end{bmatrix} \tag{4.15}$$

and $[c_x, c_y]$ (resp. $[c'_x, c'_y]$) are the co-ordinates of the centroid (or barycentrum) in the first image (resp. the second image), d and d' are defined by:

$$\begin{aligned} d &= \tfrac{1}{n\sqrt{2}} \textstyle\sum_{i=1}^{n} \sqrt{(x_i - c_x)^2 + (y_i - c_y)^2} \\ d' &= \tfrac{1}{n\sqrt{2}} \textstyle\sum_{i=1}^{n} \sqrt{(x'_i - c'_x)^2 + (y'_i - c'_y)^2} \end{aligned}$$

2. Now, the fundamental matrix $\hat{\mathbf{F}}'$ corresponding to the normalised point matches is estimated.

   (i) A linear solution for the fundamental matrix is formed from the singular vector corresponding to the smallest singular value of the SVD decomposition of the system matrix $\mathbf{Q}$ of eq. (4.14). Let $\hat{\mathbf{F}}$ be this matrix.

   (ii) Since $\hat{\mathbf{F}}$ does not necessarily obey the rank-2 constraint, this is enforced in the following manner: if the SVD of $\hat{\mathbf{F}}$ is $\mathbf{U}\ \mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)\,\mathbf{V}^T$, and $\sigma_1 \geq \sigma_2 \geq \sigma_3$, then the sought matrix $\hat{\mathbf{F}}'$ is:

$$\hat{\mathbf{F}}' = \mathbf{U}\ \mathrm{diag}(\sigma_1, \sigma_2, 0)\,\mathbf{V}^T \tag{4.16}$$

3. The matrix $\hat{\mathbf{F}}'$ is now a proper fundamental matrix. It needs to be denormalised in order to map the original image point correspondences. This is done simply [33]:

$$\mathbf{F} = \mathbf{T}'^{T}\,\hat{\mathbf{F}}'\,\mathbf{T} \tag{4.17}$$

F is then the fundamental matrix encoding the two-view geometry of the original data.

4.2.4 Recovering Euclidean motion

The main focus of this chapter is the study of linear means to recover the rigid motion of a camera from point correspondences. From the fundamental matrix, this is only possible up to a projective transformation. In other words, the estimated motion is expressed in an arbitrary projective basis. To obtain the Euclidean motion, namely the rotation matrix and the translation from the previous camera position to the current one in a Euclidean frame, it is necessary to know the camera calibration matrix **A** (see section 4.1.3). This statement is now further explained and it is shown how motion can be derived from a new matrix: the *essential matrix* **E**.

The *normalised co-ordinates* of an image point $\tilde{\mathbf{m}}$ are defined by $\tilde{\mathbf{m}_\mathbf{n}} = \mathbf{A}^{-1}\,\tilde{\mathbf{m}}$. The defining equation (4.12) of the fundamental matrix can be rewritten using normalised co-ordinates as:

$$0 = \tilde{\mathbf{m}}'^{T}\,\mathbf{F}\,\tilde{\mathbf{m}} \tag{4.18}$$

$$= \tilde{\mathbf{m}_\mathbf{n}}'^{T}\,\mathbf{A}^{T}\mathbf{F}\mathbf{A}\,\tilde{\mathbf{m}_\mathbf{n}} \tag{4.19}$$

$$= \tilde{\mathbf{m}_\mathbf{n}}'^{T}\,\mathbf{E}\,\tilde{\mathbf{m}_\mathbf{n}} \tag{4.20}$$

where **E** is called the essential matrix. Once the fundamental matrix has been computed, the essential matrix is therefore obtained by the following equation:

$$\mathbf{E} = \mathbf{A}^{T}\mathbf{F}\mathbf{A}. \tag{4.21}$$

Now, let $\tilde{\mathbf{P}}$ and $\tilde{\mathbf{P}}'$ be the projection matrices of a moving camera at two successive instants t and t'. Assume that at time instant t, the camera frame coincides with the world reference frame, the first camera projection matrix then reads $\tilde{\mathbf{P}} = \mathbf{A}\,[\mathbf{I}_3|\mathbf{0}]$. Denote the rotation matrix (resp. the translation vector) from the first to the second view by ${}^1\mathbf{R}_2$ (resp. by ${}^1\mathbf{t}_2$). The second camera matrix is thus: $\tilde{\mathbf{P}}' = \mathbf{A}\,[{}^1\mathbf{R}_2|{}^1\mathbf{t}_2]$. With this notation, it can be shown (see e.g. [32] or [27]) that:

$$\mathbf{E} = \left[{}^1\mathbf{t}_2\right]_\times {}^1\mathbf{R}_2. \tag{4.22}$$

The essential matrix has rank 2 since $\left[{}^1\mathbf{t}_2\right]_\times$ is of rank 2 and a rotation matrix is non-singular. $\mathbf{E}$ has only 5 d.o.f. and not 6 (3 for the translation plus 3 for the rotation as would have been expected). Indeed, the essential matrix is a homogeneous quantity, there is therefore a scale ambiguity. The essential matrix obeys a particular constraint which was proved by Huang and Faugeras in [38]:

A 3×3 *matrix is an essential matrix if and only if two of its singular values are equal, and the third is zero.*

This constraint is used in the following method to obtain Euclidean motion from $\mathbf{E}$ up to scale, i.e. the rotation matrix can be fully estimated, but the direction of translation alone can be extracted.

The method described here is the decomposition of the essential matrix as a product of a skew-symmetric matrix $\mathbf{S}$ and a rotation matrix $\mathbf{R}$: the so-called *RS decomposition*. If the SVD (Singular Value Decomposition [81]) of $\mathbf{E}$ is taken so that $\mathbf{E} = \mathbf{U}\,\mathrm{diag}(1,1,0)\mathbf{V}^T$, then there are two possible factorisations $\mathbf{E} = \mathbf{SR}$:

$$\mathbf{S} = \mathbf{U}\,\mathbf{Z}\,\mathbf{U}^T \qquad \mathbf{R} = \mathbf{U}\,\mathbf{W}\,\mathbf{V}^T \tag{4.23}$$

$$\text{or } \mathbf{R} = \mathbf{U}\,\mathbf{W}^T\,\mathbf{V}^T \tag{4.24}$$

where the matrices $\mathbf{W}$ and $\mathbf{Z}$ have the following expression:

$$\mathbf{W} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{Z} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.25}$$

¿From this factorisation of $\mathbf{E}$, the translation vector (up to a scale factor) can be extracted directly since $\mathbf{S} = [\mathbf{t}]_\times$. In that form, $||\mathbf{t}|| = 1$ and this unit translation vector is the third column of the matrix $\mathbf{U}$ which will be denoted by $\mathbf{u}_3$. The sign of the essential matrix cannot be determined, therefore there are two solutions for the direction of $\mathbf{t}$. There are four possible solutions for the motion. The four solutions are:

$$\mathbf{R} = \mathbf{U}\,\mathbf{W}\,\mathbf{V}^T \qquad \mathbf{t} = \quad \mathbf{u}_3 \tag{4.26}$$

$$\mathbf{R} = \mathbf{U}\,\mathbf{W}\,\mathbf{V}^T \qquad \mathbf{t} = -\,\mathbf{u}_3 \tag{4.27}$$

$$\mathbf{R} = \mathbf{U}\,\mathbf{W}^T\,\mathbf{V}^T \qquad \mathbf{t} = \quad \mathbf{u}_3 \tag{4.28}$$

$$\mathbf{R} = \mathbf{U}\,\mathbf{W}^T\,\mathbf{V}^T \qquad \mathbf{t} = -\mathbf{u}_3. \tag{4.29}$$

The ambiguity on the motion solutions is resolved by taking the solution for which the depths of the image points are all positive in both views; in other words, that the image scene must lie in front of the camera. This method is not valid if the motion undergone by the camera is a pure rotation, or if the scene is planar. In that case, the essential $\mathbf{E}$ is not defined.

In this section a simple method based on the SVD decomposition of the essential matrix to recover the motion of the camera was described. In the following section, the special case of a planar scene will be considered, and the computation of the camera motion in that case is studied.

4.3 Homography of a plane

In section 4.2, it was stated that if the observed scene was planar, it was not possible to recover the 3-D motion of the camera from the essential matrix. In fact, if

the scene is planar, there exists a simpler (and better) way to determine motion. Indeed, two images of a plane taken by a moving camera are related by a linear projective transformation called *homography* or *collineation*. The determination of this projective transformation allows the recovery of the motion of the camera [22, 86]. First, a definition of a homography is given, then a linear method to estimate it is described, and finally a method to recover the camera motion from it is presented.

4.3.1 Planar projective transformations: homographies

A homography is a *non-singular transformation of the projective plane* $\mathbb{P}^2$ *into itself.* A homography is represented by a non-singular 3×3 matrix $\mathbf{H}$ operating on homogeneous 3-vectors so that:

$$\lambda \begin{bmatrix} x'_i \\ y'_i \\ w'_i \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ w_i \end{bmatrix} \tag{4.30}$$

where $[x_i, y_i, w_i]^T$ (resp. $[x'_i, y'_i, w'_i]^T$) are the homogeneous co-ordinates of a point in the first view (resp. in the second view). The homography matrix is defined up to a scale factor λ, it has therefore 8 degrees-of-freedom which have to be identified. Four coplanar points in correspondence are required to identify the coefficients of $\mathbf{H}$, provided that no three of them are collinear.

4.3.2 Homography estimation

Estimating the homography linking two views of a scene can be performed with a linear method similar to the normalised 8-point algorithm described in section 4.2.3. Each point correspondence in the plane provides two equations:

$$\begin{cases} x'_1 \, (h_{31} \, x_1 + h_{32} \, x_2 + h_{33}) = h_{11} \, x_1 + h_{12} \, x_2 + h_{13} \\ x'_2 \, (h_{31} \, x_1 + h_{32} \, x_2 + h_{33}) = h_{21} \, x_1 + h_{22} \, x_2 + h_{23} \end{cases} \tag{4.31}$$

It is then necessary to find (at least) four point correspondences to define uniquely the transformation matrix $\mathbf{H}$ (up to a scale factor). As for the eight-point algorithm, a system matrix $\mathbf{Q}$ is built and the homography matrix is written as a 9-vector $\mathbf{h}$. $\mathbf{h}$ is solution to the homogeneous linear system $\mathbf{Qh} = 0$. If the point matches are written with their last co-ordinates equal to one, for n ($n \geq 4$) pairs of corresponding points, the system matrix $\mathbf{Q}$ ($2n \times 9$) is built in the following manner:

$$\mathbf{Q} = \begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x'_1 x_1 & -x'_1 y_1 & -x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -y'_1 x_1 & -y'_1 y_1 & -y'_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x'_n x_n & -x'_n y_n & -x'_n \\ 0 & 0 & 0 & x_n & y_n & 1 & -y'_n x_n & -y'_n y_n & -y'_n \end{bmatrix} \tag{4.32}$$

As for the eight-point algorithm, the vector $\mathbf{h}$ solution of eq. (4.32) is the singular vector corresponding to the smallest singular value of the SVD decomposition of $\mathbf{Q}$. To deal with noisy measurements, one applies the Hartley normalisation as documented in section 4.2.3.

4.3.3 Extracting motion from an homography

Let $\mathbf{\Pi} = [\mathbf{v}^T, d]^T$ be a plane in Euclidean 3-space so that a point $\mathbf{M}$ belongs to it if and only if $\mathbf{v}^T \mathbf{M} + d = \mathbf{\Pi}^T \tilde{\mathbf{M}} = 0$. The 3-vector $\mathbf{v}$ is the *normal* to the plane and d is the Euclidean distance of the plane to the centre of the world frame where every aforementioned entities are expressed. It will be shown, following [32] that the images of a plane taken from two views induce a homography and vice versa.

Given the projection matrices $\tilde{\mathbf{P}} = [\mathbf{I}_3|\mathbf{0}]$ and $\tilde{\mathbf{P}}' = [\mathbf{B}|\mathbf{a}]$ for the two views, and a plane defined by $\mathbf{\Pi}^T \tilde{\mathbf{M}} = 0$ with $\mathbf{\Pi} = (\mathbf{v}^T, 1)$, then the homography induced by the plane is $\tilde{\mathbf{m}}' = \mathbf{H}\,\tilde{\mathbf{m}}$ with

$$\mathbf{H} = \mathbf{B} - \mathbf{a}\,\mathbf{v}^T. \tag{4.33}$$

To prove this, consider the following arguments. If $\tilde{\mathbf{m}}$ is an image point in the first view, a 3-D point $\tilde{\mathbf{M}}$ lying on the corresponding ray passing through $\tilde{\mathbf{m}}$ satisfies the

relation $\tilde{\mathbf{m}} = \tilde{\mathbf{P}}\,\tilde{\mathbf{M}} = [\mathbf{I}_3|\mathbf{0}]\,\tilde{\mathbf{M}}$. Thus, any point on the ray $\tilde{\mathbf{M}} = [\tilde{\mathbf{m}}^T, \alpha]^T$ projects to $\tilde{\mathbf{m}}$, where α is a scalar parameterising the ray. This ray intersects the plane $\mathbf{\Pi}$ at the point $\tilde{\mathbf{M}}$ which satisfies $\mathbf{\Pi}^T\,\tilde{\mathbf{M}} = 0$. This yields $\alpha = -\mathbf{v}^T\,\tilde{\mathbf{m}}$. This 3-D point $\tilde{\mathbf{M}} = [\tilde{\mathbf{m}}^T, -\mathbf{v}^T\,\tilde{\mathbf{m}}]^T$ projects into the second view as

$$\begin{aligned}\tilde{\mathbf{m}}' &= \tilde{\mathbf{P}}'\,\tilde{\mathbf{M}} = [\mathbf{B}|\mathbf{a}]\,\tilde{\mathbf{M}} \\ &= \mathbf{B}\,\tilde{\mathbf{m}} - \mathbf{a}\,\mathbf{n}^T\,\tilde{\mathbf{m}} = (\mathbf{B} - \mathbf{a}\,\mathbf{n}^T)\,\tilde{\mathbf{m}}\end{aligned} \tag{4.34}$$

as required.

Now, assume that the moving camera is calibrated, and the calibration matrix is $\mathbf{A}$. If at time instant t, the camera frame coincides with the world reference frame, and at t' the motion of the camera consists of a rotation $\mathbf{R}$ and a translation $\mathbf{t}$, then the camera projection matrices read

$$\tilde{\mathbf{P}}_E = \mathbf{A}\,[\mathbf{I}_3|\mathbf{0}] \qquad \text{and} \qquad \tilde{\mathbf{P}}'_E = \mathbf{A}\,[\mathbf{R}|\mathbf{t}]. \tag{4.35}$$

Consider a world plane $\mathbf{\Pi}_E = [\mathbf{n}^T, d]^T$, so that any 3-D point $\mathbf{M}$ on the plane satisfies $\mathbf{n}^T\,\mathbf{M} + d = 0$. In that case, the homography induced by the plane is, according to 4.33:

$$\mathbf{H} = \mathbf{R} - \mathbf{t}\,\mathbf{n}^T/d \tag{4.36}$$

for the cameras $\tilde{\mathbf{P}} = [\mathbf{I}_3|\mathbf{0}]$ and $\tilde{\mathbf{P}}'_E = [\mathbf{R}|\mathbf{t}]$, and where $\mathbf{v} = \mathbf{n}/d$. Applying the transformation $\mathbf{A}$ to the images, the camera matrices $\tilde{\mathbf{P}}_E$ and $\tilde{\mathbf{P}}'_E$ are obtained and the resulting homography is

$$\mathbf{H} = \mathbf{A}\,(\mathbf{R} - \mathbf{t}\,\mathbf{n}^T/d)\,\mathbf{A}^{-1} \tag{4.37}$$

In the previous paragraphs, it was shown that a plane induced a homography and vice versa, and how the motion parameters of the camera were related to the plane geometry and the intrinsic parameters of the camera. Tsai in [86] proved that by applying a simple SVD to the homography matrix, the rotation matrix $\mathbf{R}$ and the

translation vector $\mathbf{t}$ could be easily extracted. There exist three distinct solutions to this problem depending on the multiplicity of the singular values of the SVD decomposition of $\mathbf{H}$. This method to compute motion parameters was applied in the visual servoing schemes proposed in this thesis. The method computes the SVD of the homography $\mathbf{H}_n = \mathbf{A}^{-1}\mathbf{H}\mathbf{A}$ so that $\mathbf{H}_n = \mathbf{U}\,\mathrm{diag}(\sigma_1, \sigma_2, \sigma_3)\,\mathbf{V}^T$, and the singular values are in decreasing order $\sigma_1 \geq \sigma_2 \geq \sigma_3$. Three cases arise depending of the multiplicity of the singular values:

1. The multiplicity is 1, i.e. all singular values are distinct $\sigma_1 \neq \sigma_2 \neq \sigma_3$. There are two solutions for the motion and the geometrical parameters of the plane $\mathbf{\Pi}_E$ (normal and distance to optical centre).

$$\mathbf{R} = \mathbf{U} \begin{bmatrix} \alpha & 0 & \beta \\ 0 & 1 & 0 \\ -s\beta & 0 & s\alpha \end{bmatrix} \mathbf{V}^T \tag{4.38}$$

$$\mathbf{t} = w^{-1}\,[-\beta\mathbf{U}_1 + (\frac{\sigma_3}{\sigma_2} - s\alpha)\mathbf{U}_3] \tag{4.39}$$

$$\mathbf{n} = w\,(\delta\,\mathbf{V}_1 + \mathbf{V}_3) \qquad \text{where} \tag{4.40}$$

$$\delta = \pm\sqrt{(\frac{\sigma_1^2 - \sigma_2^2}{\sigma_2^2 - \sigma_3^2})} \qquad \alpha = \frac{\sigma_1 + s\sigma_3\delta^2}{\sigma_2(1+\delta^2)} \tag{4.41}$$

$$\beta = \pm\sqrt{1-\alpha^2} \qquad s = \det(U)\det(V) \tag{4.42}$$

   where in each of the two solutions $\mathrm{sign}(\beta) = -\,\mathrm{sign}(\delta)$. $\mathbf{R}$ is the rotation matrix from the first view to the second view if $\mathbf{H}_n$ maps the first view to the second one. Similarly, $\mathbf{t}$ is the translation vector from the first to the second view, $\mathbf{n}$ is the normal to the plane. In addition, $\mathbf{U}_i$ represents the ith row of matrix $\mathbf{U}$. All entities are expressed in the camera reference frame of the first camera. Finally, w is an unknown scale factor.

2. The multiplicity is 2, i.e. two of the singular values are equal, e.g. $\sigma_1 = \sigma_2 \neq \sigma_3$, then the solution for the motion and the geometrical parameters is unique aside from a common scale factor for the translation vector:

$$\mathbf{R} = \sigma_1^{-1}\,\mathbf{H}_n - (\frac{\sigma_3}{\sigma_1} - s)\,\mathbf{U}_3\,\mathbf{V}_3^T \tag{4.43}$$

$$\mathbf{t} = w^{-1}(\frac{\sigma_3}{\sigma_1} - s)\,\mathbf{U}_3 \tag{4.44}$$

$$\mathbf{n} = w\mathbf{V}_3 \tag{4.45}$$

$$\tag{4.46}$$

where $s = \det(U)\det(V)$, and w is a scale factor. The other cases of multiplicity $\sigma_1 = \sigma_3$ and $\sigma_2 = \sigma_3$ are obtained by cyclic permutation of the indices.

3. The multiplicity is 3, i.e. $\sigma_1 = \sigma_2 = \sigma_3$, then the solution for the motion is unique: it consists of a rotation with respect to an axis passing through the centre of the first camera, there is no translation ($\mathbf{t} = \mathbf{0}$). The normal to the plane cannot be computed. The rotation matrix is unique and is given by:

$$\mathbf{R} = \sigma_1^{-1}\,\mathbf{H}_n. \tag{4.47}$$

This section shows that, for a moving camera whose intrinsic parameters are known, the relative motion arising from two different views of a plane can be extracted if at least four non-collinear points are given.

4.4 Related work

4.4.1 Camera calibration

In this chapter, the issue of calibrating the intrinsic parameters has not been covered. This falls beyond the scope of this thesis. However, intrinsic calibration is a prerequisite for every vision task requiring metric measurements (see section 4.2.4).

Until recently, camera calibration was an off-line process requiring a 3-D object with precisely known 3-D geometry, the so-called *calibration pattern*. The principle of camera calibration is simple. Given a set of image points and their corresponding 3-D points expressed in a reference frame attached to the calibration pattern, the

camera projection matrix $\tilde{\mathbf{P}}$ is estimated, and the intrinsic parameters of the camera are extracted from $\tilde{\mathbf{P}}$. Several approaches have been proposed, among which the one proposed by Tsai [85] which used a planar grid undergoing known translations, or by Robert [70] which was used to calibrate the underwater camera used in this thesis. Robert's technique requires a wedged calibration pattern with two perpendicular planes, and does not require feature extraction. Zhang recently proposed a method to calibrate a camera from a grid on a planar pattern [100], which is easier to build.

However accurate, off-line calibration techniques suffer from a major drawback. They cannot cope with varying intrinsic parameters. Mechanical and thermal variations, which are very likely to happen on an underwater camera during UUV operation, modify the intrinsic parameters. In addition, most "classical" techniques use a calibration pattern whose geometry is precisely known, such as Tsai's calibration method [85], or Robert's [70]. Autofocussing and zooming, now common features of modern cameras, are also a cause of continuous change of the parameters. Consequently, on-line camera calibration techniques which did not require any calibration pattern, have become necessary. Such techniques are referred to in the literature as *self-calibration* or *auto-calibration*. Research is still on-going on this subject (see [32], chapter 18), and the references therein for an introduction on the matter). Maybank and Faugeras proved in a theoretical paper that self-calibration was possible [53]. Later on, improvements on the method of Maybank and Faugeras were proposed by Zeller [98], and Heyden and Aström [36]. Pollefeys and Van Gool described a stratified approach to the self-calibration problem [67, 68]. From a projective reconstruction, the estimation of the homography of the plane at infinity allows the recovery of the affine structure, which is then upgraded to the Euclidean structure. Luong *et al.* derived a system of polynomial equations which, when solved, found the camera calibration parameters [48]. The aforementioned self-calibration techniques deal with constant intrinsic parameters. However, advances have been made to allow self-calibration of cameras with varying parameters. Heyden and Aström [37] proposed a method based on the assumption that the aspect ratio is known and that there is no skew to recover the focal length and the principal point [37]. Pollefeys *et al.* presented a self-calibration technique based on the assumption that

the camera was unskewed [66]. Other techniques are based on special camera motion, for instance, the works from De Agapito *et al.* was restricted to rotating and zooming cameras [14, 15].

All these techniques are performed in air; a recent paper from Lavest *et al.* [43] investigated the case of calibration underwater and showed that, for the pinhole camera model, the camera focal length in water was equal to the focal length in air times the water refractive index[1]. Therefore, it is possible to calibrate the camera in air, and from it deduce the calibration in water.

4.4.2 Epipolar geometry

Epipolar geometry is covered in many books (e.g. [21, 32, 93]); the problem of estimating the fundamental matrix has received a lot of attention, in particular a review on the subject can be found in [99]. This thesis focuses on linear estimation techniques. However, in order to obtain more accurate results, nonlinear and iterative minimisation methods are commonly used. For example, for the estimation of the fundamental matrix, the main two robust techniques (robust here is meant in a statistical sense, i.e. robust to outliers) used in computer vision are RANSAC and the Least Median of Squares (LMedS). A review of these numerical regression methods in the context of computer vision can be found in [54].

4.4.3 Homographies

Extensive coverage of homographies and their relation to the epipolar geometry is given in [32]; the recovery of motion from homographies problem was studied as early as 1982 by Tsai who proposed a solution based on the SVD of the homography matrix [86]. Later on, Faugeras in [22], and then Zhang in [101] proposed alternative proofs.

[1] See *http://www.dai.ed.ac.uk/CVonline/LOCAL_COPIES/LAVEST/main.html* for an online and shorter version of this paper.

4.5 Conclusions

The main objective of this chapter was to identify linear computer vision methods to recover the 3-D motion of a camera from two views. To achieve this objective, the concept of perspective cameras, which models satisfactorily most CCD cameras was introduced. It was then shown, using this model, that two views of a static scene were related by the epipolar geometry, and that these constraints and a knowledge of the camera's intrinsic parameters allowed the rigid motion of the camera to be computed. However, this approach fails in several cases, namely when the motion of the camera is a pure rotation, or when the imaged scene is planar. A classical technique to recover motion from a planar scene, that does not suffer from singular cases, was thus reviewed. In the following chapters, this motion recovery technique will be used for visual servoing purposes.

Chapter 5

Feature extraction and tracking

A prerequisite to sparse motion estimation techniques is the ability to find point correspondences in a couple of images. For the purpose of visual servoing however, finding point correspondences must be performed in real time, as new images are constantly being grabbed. This thesis must address the issue of *tracking* correspondences through time rather than matching them. Indeed, common methods for extraction and matching are unsuitable for our purposes because their computational burden, on standard embedded computer, exceeds "the sampling time required for successful station-keeping operation". The next two chapters will show that a sampling time of 200 ms was sufficient to achieve successful station-keeping of a typical ROV.

One of the goals of this thesis was the design of an automatic station-keeping algorithm for UUVs. Consequently, the designed algorithm must also be able to work on real underwater images.

A tracking algorithm, suitable for the application, i.e. able to run on real underwater images at a 5 Hz sampling rate, is thus presented [28, 83] in this chapter, and an experimental evaluation of its performances is carried out.

5.1 Introduction

Most visual control techniques in the literature are based on the extraction of image features such as points, lines, circles or ellipses [20, 39]. Finding these image features proves difficult, in particular in unstructured and unknown scenes. Researchers have therefore reduced the image processing load by applying their techniques to simple and dedicated visual targets (see e.g. figure 5.1). The image processing is then

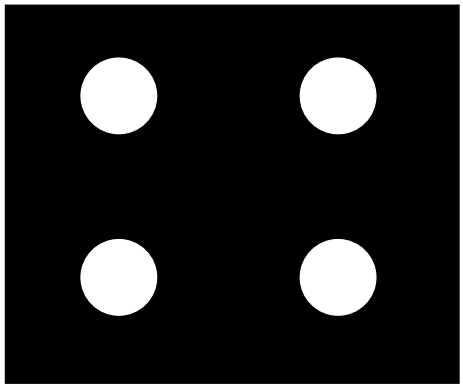


Figure 5.1: A simple visual servoing target.

reduced to a thresholding and the matching is straightforward. In this thesis, the more complex task of working on "natural" images such as the one depicted in figure 5.2 was tackled; the ultimate goal being to perform visual station keeping at sea, near the sea bottom for instance. Recent visual servoing work on natural images include that of S. Negahdaripour [61] and A. Crétual [10]. In order to estimate the relative motion of the scene with respect to the camera, they applied *optical flow* techniques. The optical flow is an estimate[1] of the 2-D motion field based solely on the data available in images, that is the spatial and temporal variations of image brightness. The 2-D *motion field* is the perspective projection onto the image plane of the true 3-D velocity field of moving surfaces in space [87]. Optical flow is a dense motion estimation technique: it is estimated at each point in the images. Consequently, this approach is computationally expensive, and images have to be reduced in size to allow real-time performances. For instance Negahdaripour used 64×64 images, and Crétual restricted the search area by applying a binary mask to the image for his pedestrian tracking application.

In this thesis however, experiments have been carried out on full resolution 512×512 images, since a sparse feature tracking method was employed. Reducing the number of features to track allows faster computation time. However, one still has to search for feature points correspondences in the whole image contrary to stereo applications. In stereo, the knowledge of the rigid transformation between the two calibrated

[1] It is often assumed that the optical flow and the motion field coincide, however this is seldom the case [87].

Figure 5.2: A "natural" image: picture of the ship wreck Kirk Pride taken from a manned submarine at 250 metre depth. Courtesy of Philip Greenspun, M.I.T., *http://www.photo.net/webtravel/cayman*.

cameras reduces the search from 2-D to 1-D, one point in the first image lies on an epipolar line in the second image. Several methods have been devised to reduce the search area from the whole image to a more tractable one. One method consists in assuming that the motion between consecutive images is sufficiently small to restrict the search in a window around the previous feature position. Applying prediction schemes on the features' motion also produces faster searches. Prediction schemes such as $\alpha - \beta$ filters (constant velocity filters) [8, 74], or Kalman filters [1] can be used. The tracking approach which will be introduced in this chapter assumed that the motion between consecutive images is sufficiently small. The originality of the approach of Shi, Tomasi and Kanade [75, 82] was to extract, from an image, features which satisfied properties guaranteeing that they would suitably be tracked in subsequent images through time. This idea allowed the selection of "good tracking features".

This type of approach was used in the visual servoing scheme of chapter 7. The original algorithm from Shi, Tomasi and Kanade was later improved by Trucco *et al.* in [83] with an automatic thresholding scheme.

The details of the modified algorithm will be described in section 5.2. A quantitative

performance evaluation of the latter in underwater conditions is reported in section 5.3.

5.2 The feature tracker

The feature implementation described in this thesis is a modification of the Shi-Tomasi-Kanade tracker [75, 82] which employs an automatic outlier rejection scheme [28, 83]. In this thesis' context, *robust tracking* means to automatically detect unreliable matches or *outliers* [71].

The main characteristics of this tracker are the following:

- Features are extracted on the basis of their suitability for tracking: the selected features are the ones which will be easily tracked.
- The tracking, i.e. the estimation of the motion field for each selected feature between two consecutive frames, follows a pure translation motion model (although an affine motion model could be used too).
- Outlier rejection: if a tracked feature is too dissimilar between the first frame and the current frame, it is discarded as a bad feature or *outlier*. This selection is based on an affine motion field model, since too many features would be discarded if a pure translational model was used.

5.2.1 Motion field models

These models rely on the basic assumption that intensity values within small regions remain practically unchanged after small displacements. If $I(\mathbf{x}, t)$ represents an image sequence, with $\mathbf{x} = [u, v]^T$ a point in the image and t time, the above assumption reads:

$$I(\mathbf{x}, t) = I(\delta(\mathbf{x}), t + \tau) \tag{5.1}$$

where $\delta(\cdot)$ is the motion field. For an *affine motion model*, which takes into account translation and scaling, the motion field is:

$$\delta(\mathbf{x}) = \mathbf{D}\,\mathbf{x} + \mathbf{d}. \tag{5.2}$$

where

$$\mathbf{D} = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \tag{5.3}$$

$\mathbf{D}$ is a deformation matrix and $\mathbf{d}$ is the translation of point $\mathbf{x}$. If a point $\mathbf{x}$ in image I moves to the point $\mathbf{A}\,\mathbf{x}+\mathbf{d}$ in image J, then, omitting the time variable for clarity,

$$J(\mathbf{A}\,\mathbf{x} + \mathbf{d}) = I(\mathbf{x}) \tag{5.4}$$

where $\mathbf{A} = \mathbf{I}_2 + \mathbf{D}$ and $\mathbf{I}_2$ is the 2×2 identity matrix. The motion parameters sought, $\mathbf{D}$ and $\mathbf{d} = [d_1, d_2]^T$, are obtained by minimising the dissimilarity measure ϵ on a chosen window W:

$$\epsilon = \sum_W [I(\mathbf{x} + \mathbf{d}, t + \tau) - I(\mathbf{x}, t)]^2. \tag{5.5}$$

Shi and Tomasi [75] showed that, approximating the squared differences by their first-order Taylor expansion, solving eq. (5.5) is equivalent to finding the vector $\mathbf{z} = [d_{11}, d_{12}, d_{21}, d_{22}, d_1, d_2]^T$ solution of the linear system

$$\mathbf{T}\,\mathbf{z} = \mathbf{a}, \tag{5.6}$$

in which, if the partial derivation of I with respect to a variable x is denoted by $I_x = \partial I/\partial x$,

$$\begin{aligned} \mathbf{a} &= -\tau \sum_W I_t \,[uI_u, uI_v, vI_u, vI_v, I_u, I_v]^T, \\ \mathbf{T} &= \sum_W \begin{bmatrix} \mathbf{U} & \mathbf{V} \\ \mathbf{V}^T & \mathbf{Z} \end{bmatrix} \end{aligned}$$

with

$$
\begin{aligned}
\mathbf{U} &= \begin{bmatrix} u^2 I_u^2 & u^2 I_u I_v & uv I_u^2 & uv I_u I_v \\ u^2 I_u I_v & u^2 I_v^2 & uv I_u I_v & uv I_v^2 \\ uv I_u^2 & uv I_u I_v & v^2 I_u^2 & v^2 I_u I_v \\ uv I_u I_v & uv I_v^2 & v^2 I_u I_v & v^2 I_v^2 \end{bmatrix}, \\
\mathbf{V}^T &= \begin{bmatrix} u I_u^2 & u I_u I_v & v^2 I_u^2 & v I_u I_v \\ u I_u I_v & u I_v^2 & v I_u I_v & v I_v^2 \end{bmatrix}, \\
\mathbf{Z} &= \begin{bmatrix} I_u^2 & I_u I_v \\ I_u I_v & I_v^2 \end{bmatrix}
\end{aligned}
$$

These equations allow the estimation of the affine motion field, and are used to discard possible outliers. However, under the assumption that the motion between two frames is sufficiently small (a few pixels), a simpler translational motion model is adequate.[2] Indeed, not only is a translational model computationally cheaper, but it is also preferable to the affine model since for small motions the deformation matrix $\mathbf{D}$ is difficult to estimate accurately [75]. In that case, the deformation matrix is equal to zero, and the motion field is $\delta = \mathbf{d}$, and is estimated by solving a linear system:

$$\mathbf{G}\,\mathbf{d} = (\sum_{W} \mathbf{Z})\,\mathbf{d} = \mathbf{e} \tag{5.7}$$

where the residual $\mathbf{e}$ of a given feature, is made up of the last two entries of $\mathbf{a}$:

$$\mathbf{e} = -\tau \sum_{W} I_t \,[I_u, I_v]^T.$$

To summarise, the motion of each extracted feature between two consecutive frames is estimated according to eq. (5.7), assuming small displacements. The affine motion model is used for larger displacements which do not satisfy the translational model.

[2] In practice, the vision system must be able to process images at a sufficiently high frame rate to guarantee that the assumption holds.

5.2.2 Feature extraction

Features for which eq. (5.7) yields a numerically stable solution for $\mathbf{d}$ are good features. This is the case if both eigenvalues λ_1 and λ_2 of $\mathbf{G}$ are well above noise level and are not too dissimilar. In practice, if $\min(\lambda_1, \lambda_2) > \lambda$ where λ is a user-defined threshold[3], eq. (5.7) is taken to be sufficiently well-conditioned. In other words, the goal is to extract features that can be tracked reliably through time.

5.2.3 Automatic outlier rejection

This section summarises the automatic outlier rejection strategy. The reader is referred to [28] for a complete mathematical treatment. To detect outliers automatically, the tracker assumes that between the first frame and the current frame the illumination of the windows is constant, and the difference in intensities is caused by centred Gaussian distributed noise. Fusiello *et al.* [28] showed that the residuals ϵ_i of the n tracked features, obtained with the affine motion model, follow a Gaussian distribution, provided that the window size was at least 7×7 (pixels). If a residual was not a sample from the Gaussian distribution, it was an outlier, in which case the corresponding feature was discarded. The criterium (X84) used to decide whether a residual belonged to the distribution was the Median Absolute Deviation (MAD): $\text{MAD} = \underset{i}{\text{med}} \{|\epsilon_i - \underset{j}{\text{med}}\, \epsilon_j|\}$ where $\text{med}\,\{\}$ is the median (see Rousseeuw's book for details [71]). The X84 rule prescribed rejecting the values which were k median absolute deviations away from the median. A value of $k = 5.2$, under the hypothesis of a Gaussian distribution, was adequate in practice since it corresponded to about 3.5 standard deviations, and this range around the mean value of the distribution contained more than 99.9 % of the distribution. In addition, the *breakdown point* of the X84 rule was 50 %, which means that any majority of the data overruled any minority.

Features extracted from the first frame were guaranteed to be tracked reliably in

[3] In this thesis' implementation, the threshold was implicitly defined by the number of features wished to be obtained at the beginning, which depended on the requirements of the visual servoing application. For example, estimating homographies requires at least four features.

subsequent frames under a translational model of image motion. The displacement of a given feature between two frames was approximated by a translation vector **d**. To ensure that the tracked features were still reliable at a given point in time, an affine motion model was fitted to each feature between the first frame and the current frame. If the residual of this feature was not a sample of the Gaussian distribution of the residuals of all features, it was discarded.[4]

5.3 Performance evaluation of a feature tracker for underwater applications

In order to correctly evaluate the causes of possible visual servoing experiments failures of chapter 7, it was necessary to estimate the feature tracker performances and limitations in the same operating conditions. Visual servoing was tested in an underwater test tank, filled with fresh water, at a short range of 1–2 metres. The visual servoing schemes were run at a sampling rate of 5 Hz on a non-real-time operating system.[5]

Characterising feature trackers proves difficult since performance depends on several factors such as:

- the nature of the imaged scene (texturedness, albedo, etc.);
- the optical properties of the medium, in this case, water, and in particular, the absorption and diffusion of the light;
- the nature of light sources, whether it is natural (sunlight filtering from the surface) or artificial (on-board underwater lights), or a combination of these.

[4] During this PhD work, a real-time version of the automatic outlier rejection was unfortunately not available. The feature tracker evaluated underwater was therefore equivalent to the Shi-Tomasi-Kanade tracker.

[5] A *real-time operating system* is meant to be an *operating system where it is possible to guarantee that events will happen at a definite time within known and certified accuracy.* For example, a typical robot controller will have a sampling rate of 5 ms, and new control inputs would be guaranteed to happen every 5 ms $\pm$ 5 μs. For a non real-time operating system, such events may happen every 5 ms, however it may sometimes happen a few seconds later!

Consequently, an exhaustive study would be too time consuming and too difficult to achieve. An evaluation of the following characteristics, relevant to the overall purpose of this thesis, was carried out:

- maximum admissible interframe displacements,
- influence of window size on tracking performance,
- tracking consistency (to be defined more precisely),
- sensitivity to illumination conditions.

5.3.1 Experimental setup

The following experiments used a black and white CCD Mariscope camera "Micro" placed into a waterproof housing (see figure 5.3). Its light sensitivity was rated at 0.1 Lux by the manufacturer, and its horizontal viewing angle was 43°, which meant that if the camera was placed at one metre above the bottom of the tank, the area of the imaged scene was roughly $1 \times 1\, m^2$.

For the purpose of the experiments, the camera was rigidly mounted at the end of a pole along the Z-axis of the Cartesian robot (see figure 5.4). The camera was set up so that the image plane was parallel to the flat bottom of the tank. The camera co-ordinate frame was parallel to the Cartesian robot co-ordinate frame. Consequently, the relative motion of the camera and the relative motion of the robot could be assumed to be identical, to within the vibrations of the camera at the end of the pole, which could not be measured.

A 150-watt underwater light was placed beside the camera to simulate a typical underwater vehicle lighting configuration (see figure 5.4).

To be able to compare the tracker's performance in the same conditions as during the visual servoing experiments, the tests were run at the same rate, i.e. five 512×512 pixels 8-bit images were grabbed and processed per second. The search window size W was set to 13×13 pixels in all experiments. Twenty features were extracted in the first frame, of which a few were systematically lost when they came out of

the field of view during the motion experiments. These features were therefore not taken into account to estimate the tracker's performance.

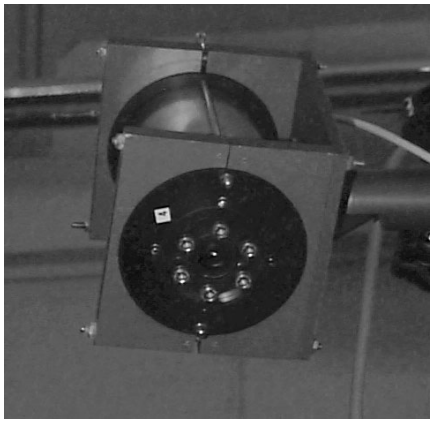


Figure 5.3: Underwater camera "Micro" from Mariscope in its original housing.

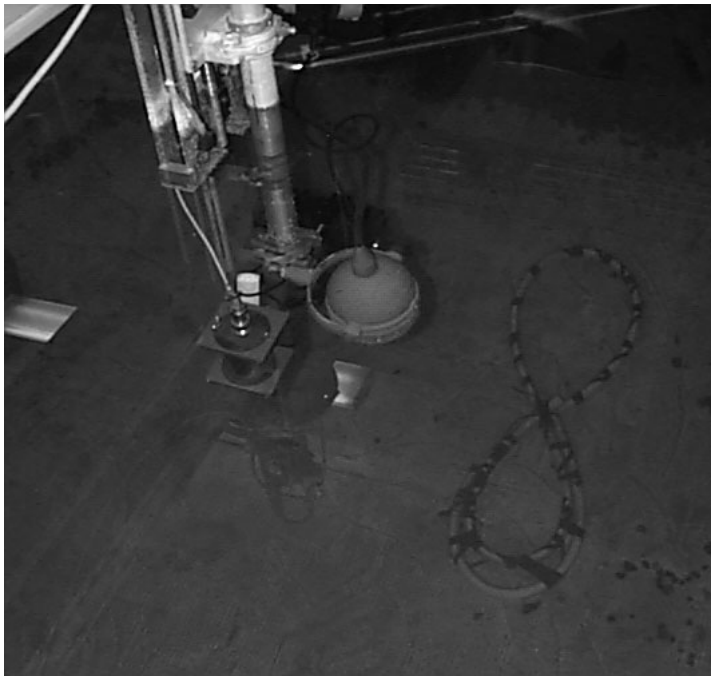


Figure 5.4: Experimental setup used to evaluate the tracker's performance. The underwater light and the camera were fixed at the extremity of the Cartesian robot.

5.3.2 Interframe displacements characterisation

This experiment studied the relationship of the number of tracked features through time with respect to the velocity of the camera. The tracker was expected to lose

features more quickly as the average pixel displacement between consecutive frames was increased. To study this relationship, the Cartesian robot was moved back and forth at various constant speeds. The amplitude of motion was constrained so that roughly a third of the original image always stayed within the camera's field of view, which meant that the robot was moving no more than 15 cm away from the starting position. Figure 5.7 illustrates a typical motion of the robot for each axis (X and Y), while figure 5.5 pictures the robot axes. For each robot's velocity value v_i (1 cm/s, 2 cm/s, 3 cm/s, 4 cm/s, and 5 cm/s), a set of experiments was performed with four different velocity angles α_k (30°, 45°, 60° and 135°), so that the robot's velocity vector $\mathbf{v}_i$ was $[v_i \cos(\alpha_k), v_i \sin(\alpha_k)]^T$. For each of the twenty experiments, the number of tracked features was monitored. Figure 5.8 shows the average number of features over each velocity angle for each velocity value. The experiments were stopped as soon as the number of tracked features fell below four.

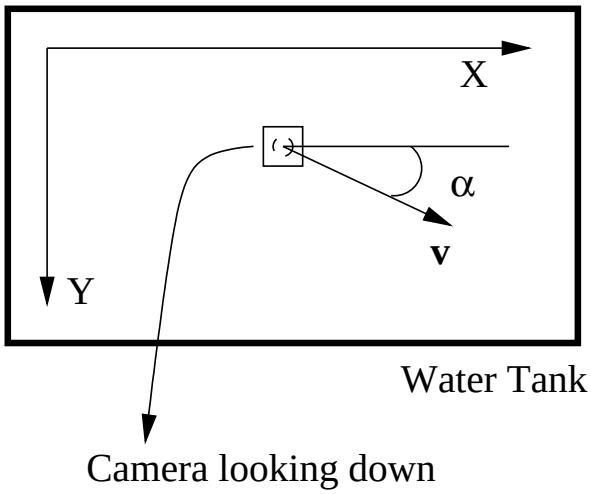
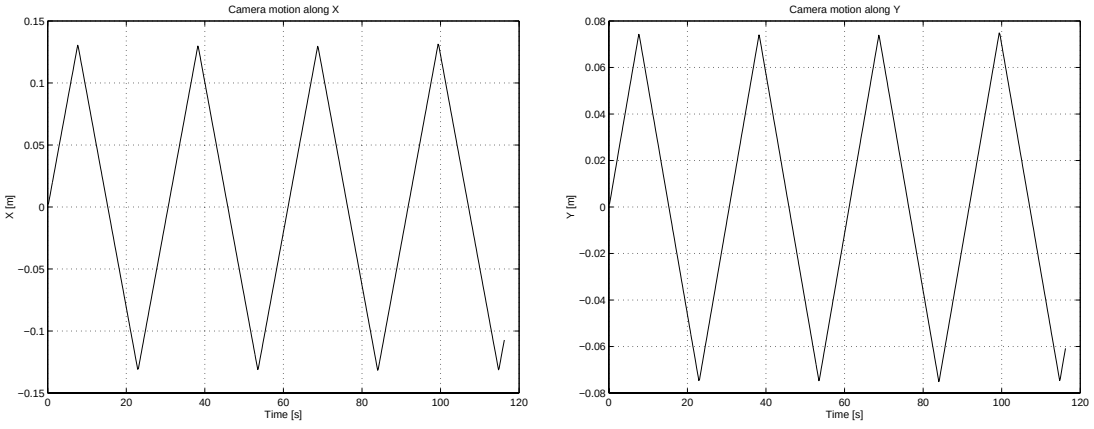


Figure 5.5: Camera motion in the water tank with axes: top view.

Overall, the number of features successfully tracked decreased markedly as speed increased. The tracking was best at a velocity of 3 cm/s, contradicting the general trend. However, there was a noticeable performance drop for velocities greater than 3 cm/s. Note that the first few features were lost because they were moving out of the field of view. After one motion cycle though, all the lost features were associated to the tracking algorithm. Thus, no statistics on the number of features itself were attempted.

Figure 5.6: Overview of the water tank: Cartesian robot, underwater camera and light.



(a) (b)

Figure 5.7: Interframe displacements characterisation: (a) Camera's motion along the X-axis. (b) Camera's motion along the Y-axis.

The focal length of the camera was $(f_x, f_y) = (801.6, 787.0)$ (see table 5.1). Therefore, with the bottom of the tank being 1.1 m away from the focal plane, one pixel in the image corresponded to a square of 1.4 mm sides at the bottom of the tank. Consequently, and according to the results plotted in figure 5.8, the tracker could track features for an extended amount of time as soon as the average interframe pixel displacement was less than 4 pixels. When only 4 features remained, the tracking algorithm was stopped intentionally.

These experiments gave an approximate idea of the limits of the interframe displacement that could be measured by the feature tracker in a difficult underwater situation. In the computer vision field, it is difficult to assess an algorithm's performance on real data and to deduce, from the results on the necessary limited set of real data, its performance in different conditions. The experiments in this thesis were chosen to be indicative and realistic.

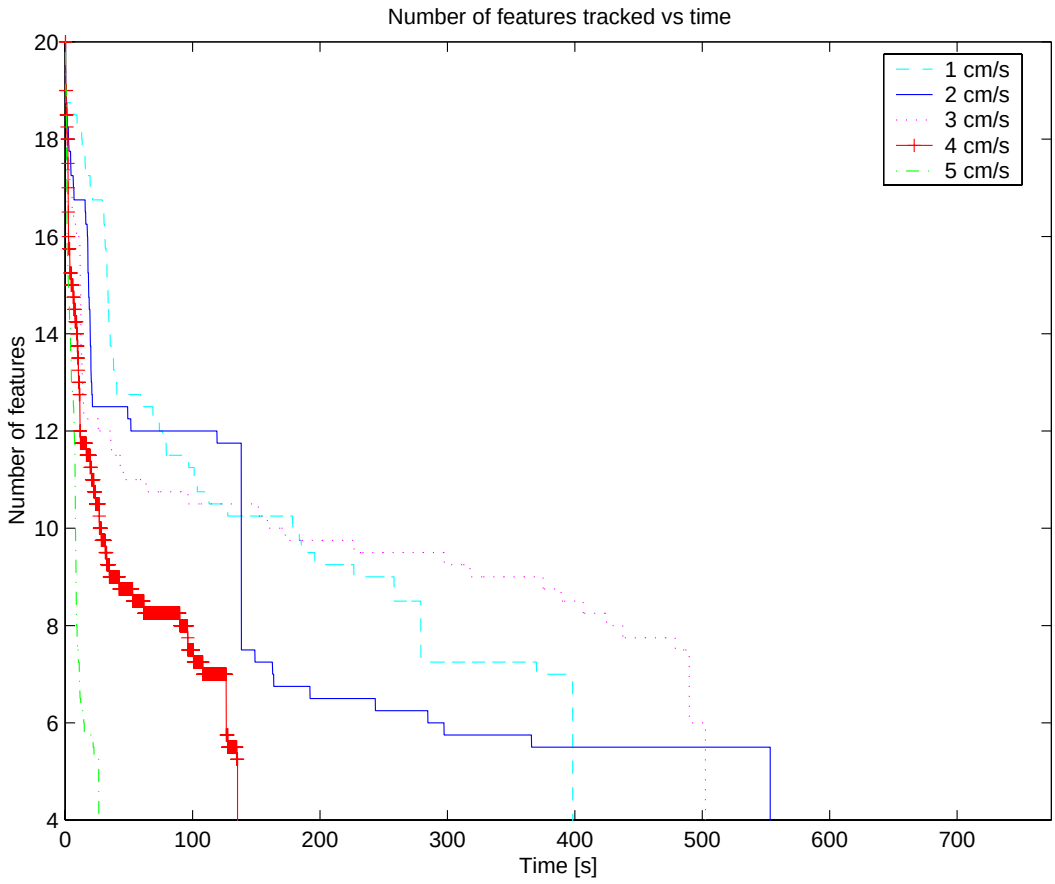


Figure 5.8: Average number of tracked features vs time for different camera velocities.

α_x	801.6 $\pm$0.4 pix
α_y	787.0 $\pm$0.4 pix
s	0.0 pix/m
x_0	292.1 $\pm$0.6 pix
y_0	242.6 $\pm$0.7 pix

Table 5.1: Camera calibration parameters underwater. Data courtesy of Christophe Leperon, LASMEA, France.

5.3.3 Influence of extraction and search window size

This section studies the influence of the size of the extraction and search window on the interframe displacement measurement capability of the feature tracker. The same experiments as the ones mentioned above were performed at a given speed for a set of window sizes.

The first experiments moved the camera at the observed optimal speed of 0.3 cm/s with an angle of 45 $^\circ$. The evolution of the number of tracked features for each window size through time (from 9×9 pixels to 25×25) is illustrated by figure 5.9. An optimal result was obtained in this experiment with a window's size of 21×21 pixels.

These results were also plotted in a different way, figure 5.10. Here the number of features tracked through time have been integrated and normalised by dividing the result of this integration by the maximum value at the last iteration.

In other words, if $n_i(k)$ was the number of features still tracked by the feature tracked at iteration k, for window size W_i, the integral $N_i(k)$ plotted was:

$$N_i(p) = \frac{1}{M} \sum_{k=1}^{k=p} n_i(k) \tag{5.8}$$

where M was the maximum value over all N_i at the last iteration. It is clear that the best tracking results were obtained for a window size of 21×21.

The above results imply that choosing an extracting and search window of size 21×21 would provide the best tracking performance in all conditions. As mentioned earlier,

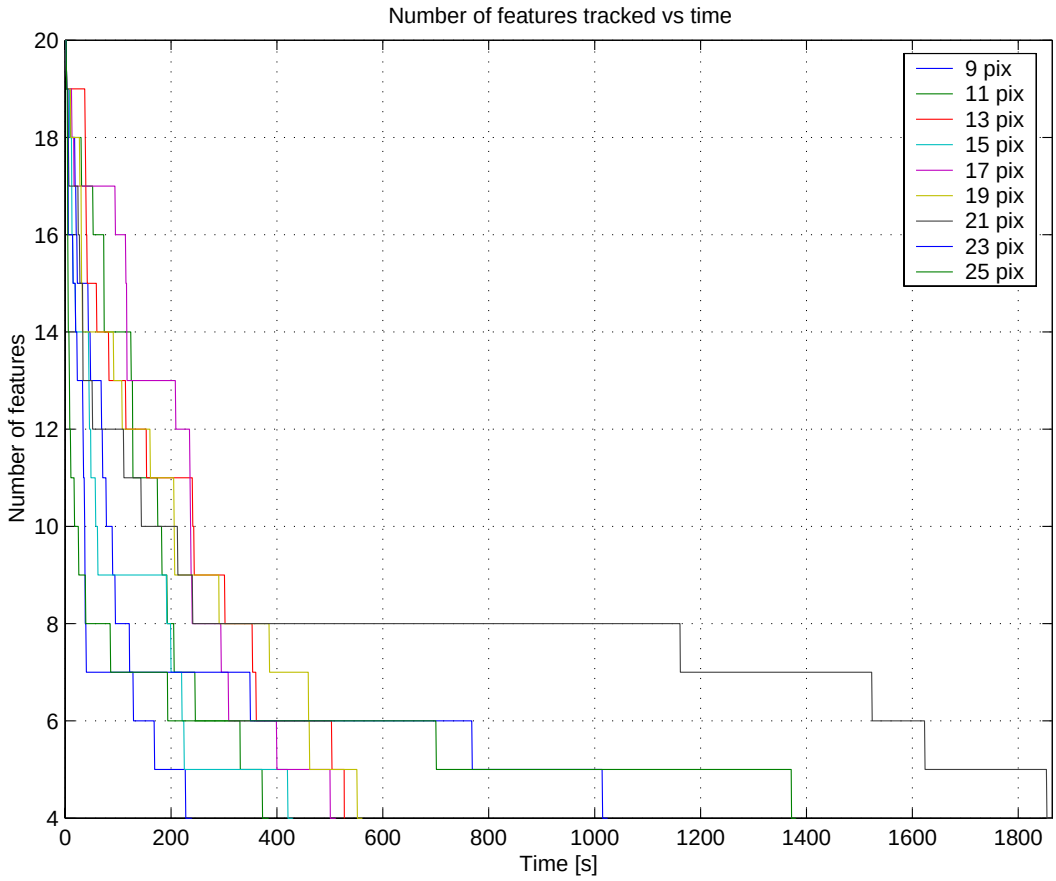


Figure 5.9: Number of tracked features vs time for different window's sizes with a camera velocity of 0.3 cm/s. An optimal result was obtained in this experiment with a window's size of 21×21 pixels.

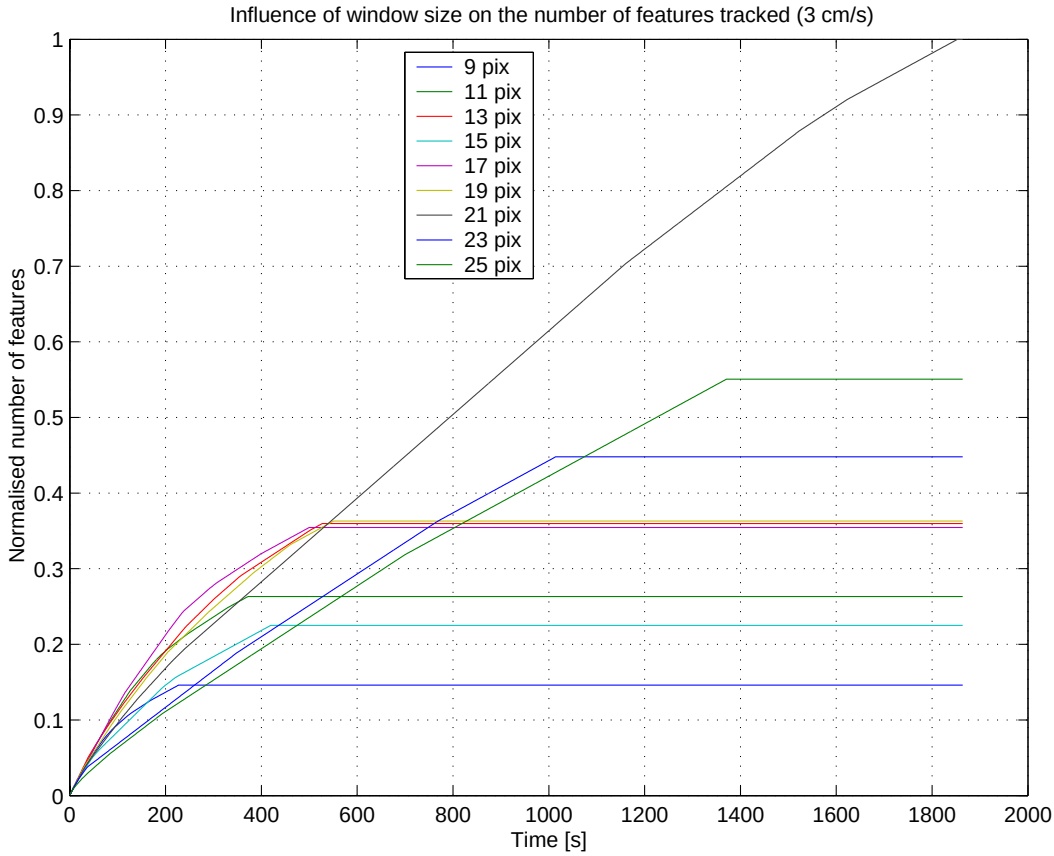


Figure 5.10: Integrated normalised number of tracked features vs time for different window sizes with a camera velocity of 0.3 cm/s. An optimal result was obtained in this experiment with a window size of 21×21 pixels.

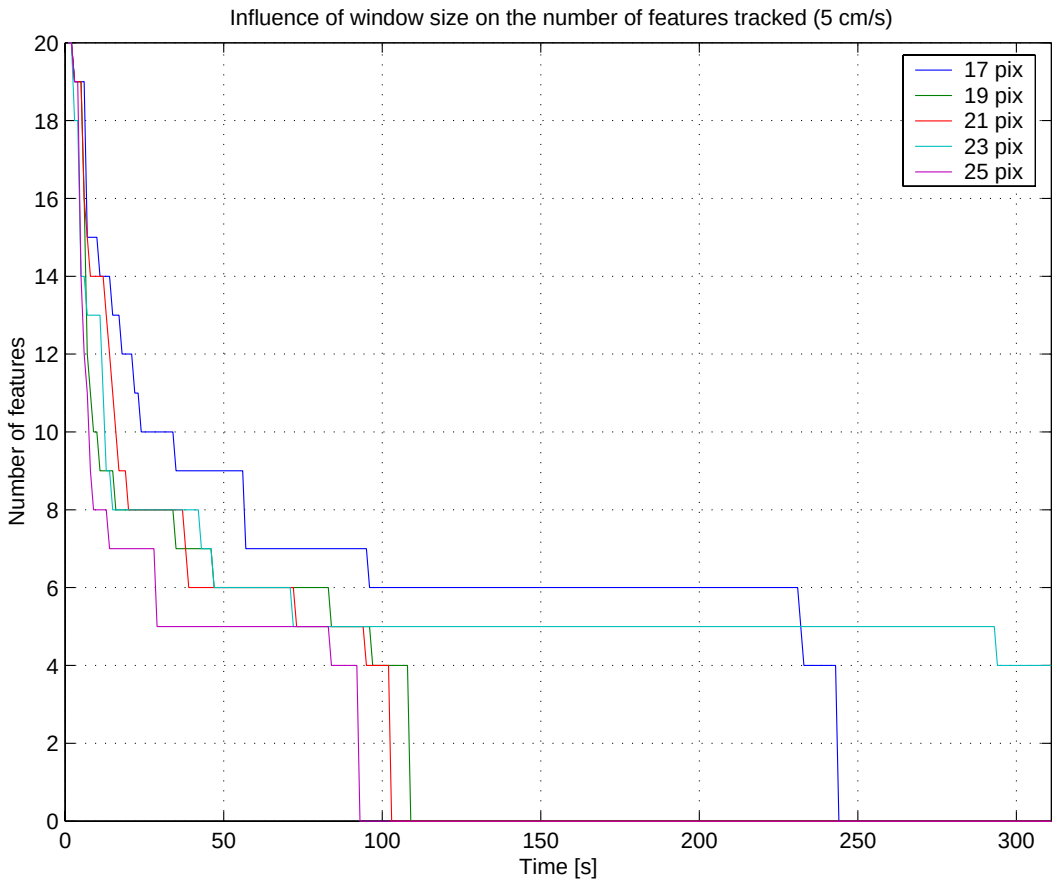


Figure 5.11: Number of tracked features vs time for different window sizes with a camera velocity of 0.5 cm/s. An optimal result was obtained in this experiment with a window size of 17×17 pixels.

vision research suffers from the difficulty to infer general rules on an — always limited — number of real world experiments. Indeed, the same experiment, at a higher speed of 0.5 cm/s found that the optimal window size value was 17×17 instead! Figures 5.11 and 5.12 clearly illustrate that result.

The main conclusion to take from these experiments is that the window size has an influence on the tracking performance but that this influence was not quantifiable, at least not with only those two experiments. It is also very likely that the results obtained here would change if the nature of the imaged scene changed. During these experiments, the bottom of the tank was covered with gravel. The tank had just been filled up. Therefore lots of dust particules were cloudying the water, altering the visibility conditions.

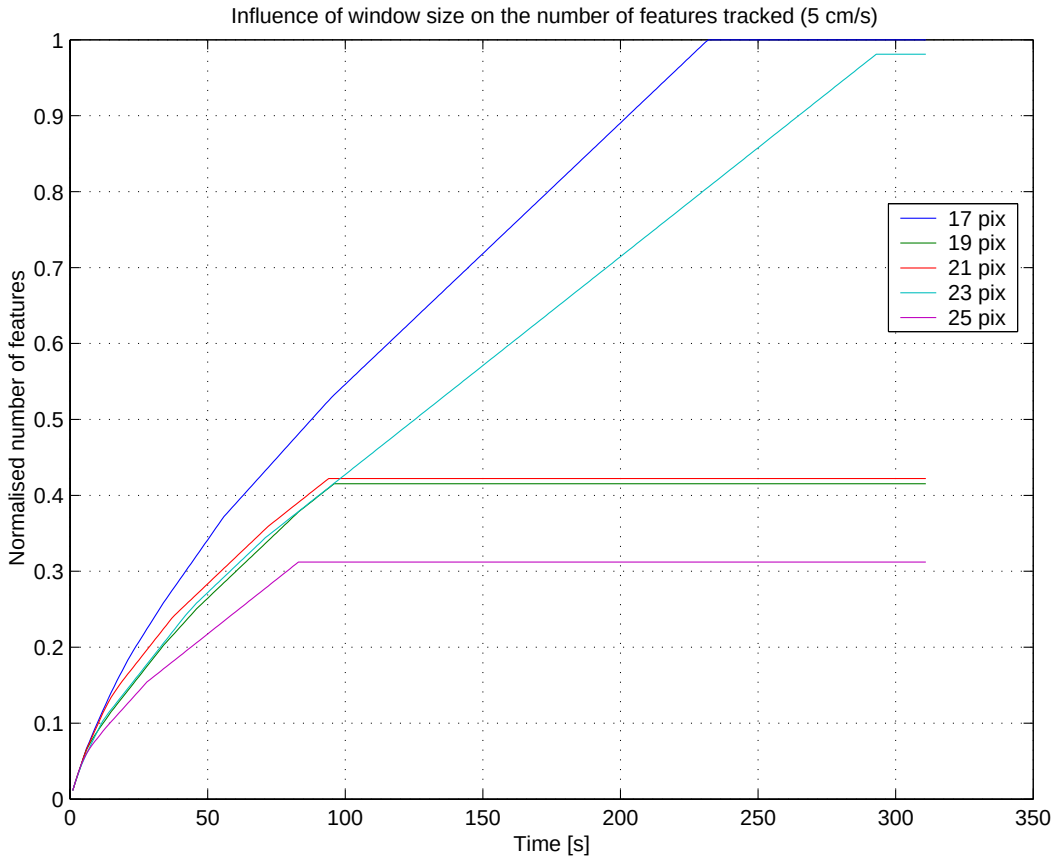


Figure 5.12: Integrated normalised number of tracked features vs time for different window's sizes with a camera velocity of 0.5 cm/s. An optimal result was obtained in this experiment with a window size of 17×17 pixels.

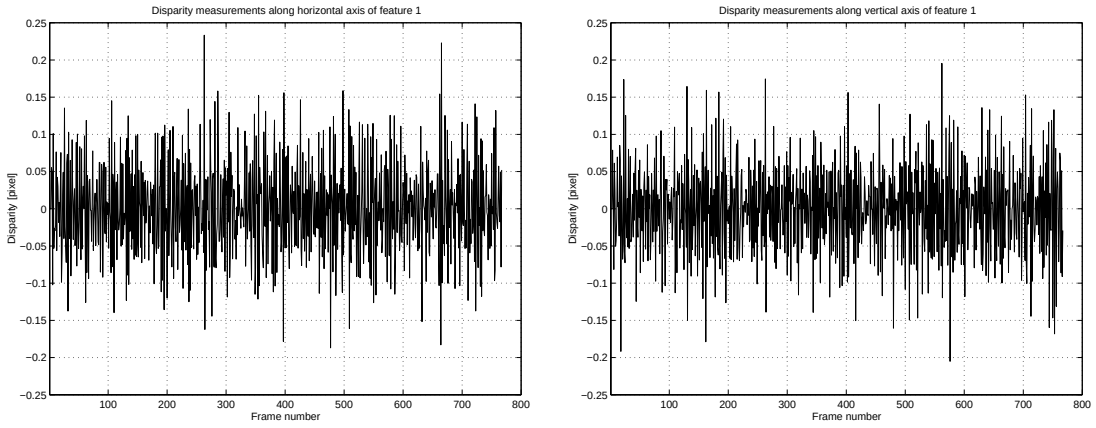
5.3.4 Consistency of tracking

A feature tracker is by nature a velocity sensing device. The noise affecting the CCD camera, the lighting conditions and the digitisation will therefore affect the interframe displacement measurements between two consecutive frames. The tracking will be consistent if extracted features are reliably found by the tracker at their initial location when there is no motion in the image.

In order to characterise the consistency of the tracking, some static tests were first run. The camera was immobile, looking at a static underwater scene. Features were extracted and subsequently tracked for roughly 1,000 frames. Ideally, the measured displacement between two consecutive frames should have been zero. Assuming that the tracking algorithm was affected by Gaussian noise, the consistency of tracking in the image can be characterised by its mean μ and its standard deviation σ. A typical consistency plot is shown in figure 5.13.

A series of static tests was performed on 14 different locations of the test tank with different types of scenes including: gravel, a concrete bottom, a concrete block, and a yellow plastic tube. For each location, 19 features were tracked for about 1,000 frames. The mean displacement between consecutive frames and its associated standard deviation over the whole sequence and over all features were calculated. The results are summarised in table 5.2. The computed mean displacements were very small, the maximum value was of the order of 10^{-3} pixel. These mean values were statistically meaningful since the maximum standard error of the mean: $\epsilon = \sigma/\sqrt{n}$, where $n = 1000$ samples had a maximum value of $3.\,10^{-3}$ pixel. It can therefore be safely assumed that under good lighting conditions, there was a bias smaller than one hundreth of a pixel in the measurement of displacement between frames. This means that a feature can be located consistently within 0.03 pixels 99.7 % of the time ($3\,\sigma$ rule).

Similar tests were also performed for varying lighting conditions. These tests are described in the next section.



(a) **(b)**

Figure 5.13: Typical displacement measurements of one feature in the image when the camera was not moving: (a) Along the horizontal axis; (b) Along the vertical axis.

	Mean [pixel]		Standard deviation [pixel]	
Scene type	Horizontal	Vertical	Horizontal	Vertical
Concrete block	$-2.3\,10^{-4}$	$1.1\,10^{-4}$	0.0837	0.0614
Concrete bottom 1	$-4.7\,10^{-4}$	$-0.9\,10^{-4}$	0.1312	0.0727
Concrete bottom 2	$0.1\,10^{-4}$	$-0.8\,10^{-4}$	0.1251	0.0806
Gravel 1	$-1.0\,10^{-4}$	$-0.2\,10^{-4}$	0.0888	0.0599
Gravel 2	$-0.5\,10^{-4}$	$-0.5\,10^{-4}$	0.0872	0.0662
Gravel 3	$15.0\,10^{-4}$	$43.0\,10^{-4}$	0.1275	0.1171
Gravel 4	$-0.2\,10^{-4}$	$-0.4\,10^{-4}$	0.0889	0.0701
Gravel 5	$2.0\,10^{-4}$	$0.0\,10^{-4}$	0.0780	0.0539
Gravel 6	$0.0\,10^{-4}$	$-1.0\,10^{-4}$	0.0837	0.0589
Gravel 7	$-1.9\,10^{-4}$	$-0.7\,10^{-4}$	0.0818	0.0597
Tube 1	$1.5\,10^{-4}$	$-4.0\,10^{-4}$	0.0811	0.0606
Tube 2	$1.0\,10^{-4}$	$0.3\,10^{-4}$	0.0819	0.0574
Tube 3	$9.9\,10^{-4}$	$10.0\,10^{-4}$	0.0846	0.0610
Tube 4	$0.3\,10^{-4}$	$1.2\,10^{-4}$	0.0780	0.0740

Table 5.2: Mean displacement and its associated standard deviation for a set of 14 static tests in the water tank.

5.3.5 Sensitivity to illumination

To evaluate the sensitivity of the feature tracker to illumination, the following tests were performed. The camera was made to look downwards to the bottom of the water tank. The latter was covered with gravel and other objects as can be seen in figure 5.14. The camera was attached to the Cartesian robot. An underwater light was placed beside the latter (see figure 5.4). The input voltage to the light could be adjusted by means of a variable transformer. The results shown here were relative to this input voltage, since a luxmeter sensitive enough to measure the actual illumination of the scene was not available. Indeed, the available luxmeter could not measure illumination below 1 lux. The underwater camera had a rated sensitivity of 0.1 lux.

Two sets of experiments were performed. In the first one, the scene was mostly composed of gravel; in the second, a marked hose pipe was used, as in some of the author's early visual servoing experiments [46]. Figure 5.14 shows the two underwater test scenes with the same illumination level. All runs were 1,000 frames long.[6]

Gravel scene

This test was carried out with illumination levels ranging from 30 % to 55 %. Below 30 % and over 55 %, the tracker did not run for a meaningful amount of time. Figures 5.15 and 5.16 illustrate the tracking consistency with respect to illumination conditions. The mean displacement for the range 30–55 % was very close to zero (well within a 3σ bound), which meant that in these conditions, the tracking was not biased. However, for the 30 % illumination level, the tracker lost track very quickly (279 frames). The study of the standard deviation graph illustrates clearly the improvement over tracking consistency and hence on the overall performance of the tracker, as the scene is better lit. The relationship seemed to be exponential on the studied range. Note that if the scene was too lit, the tracker failed very quickly.

[6] Except for two extremes illumination cases where all features were lost before 1,000 frames were grabbed (gravel scene with 30 % illumination level, and tube scene with 55 % illumination level).

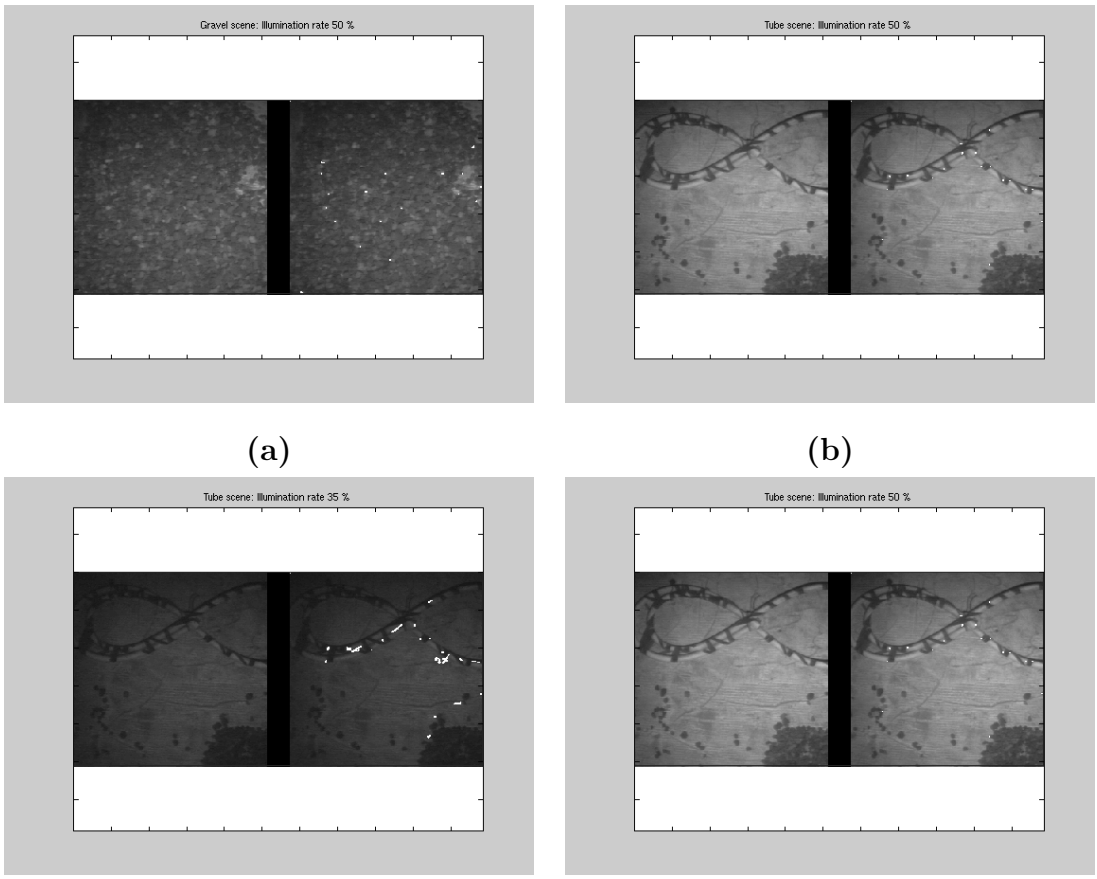


Figure 5.14: Two underwater scenes under the same illumination: (a) Gravel with a 50 % illunination level, (b) Tube with a 50 % illumination level. Note the difference of contrast due to the nature of the imaged scene. The "tube scene" under two illumination levels: 35 % (c) and 50 % (d). The difference in contrast is caused here by the amount of light dispensed to the scene.

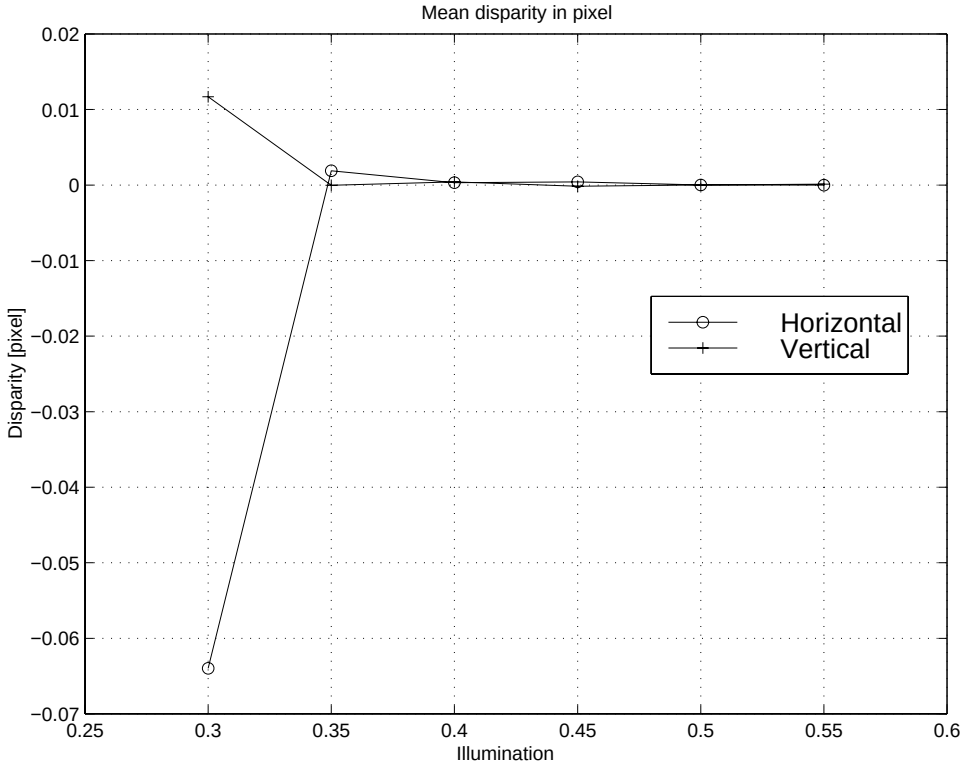


Figure 5.15: Average displacement measures with respect to illumination conditions on a gravel scene.

Tube scene

The "tube scene" being much brighter than the gravel scene, the tracker failed more quickly, namely at a 55 % illumination level.[7] This shows the necessity of measuring the illumination reflected by the scene, and not the light dispensed.

Conclusions

In reasonably well-lit conditions, the feature tracker was able to track features consistently to within 0.5 pixel standard deviation and was not biased (zero mean). This will turn into possible drift (random walk) on the position measurements through integration of tracking errors through time.

Further tests with a more sensitive luxmeter should be carried out to derive the relationship between the tracker consistency and the actual illumination of the scene.

[7] This test was 150 frames long.

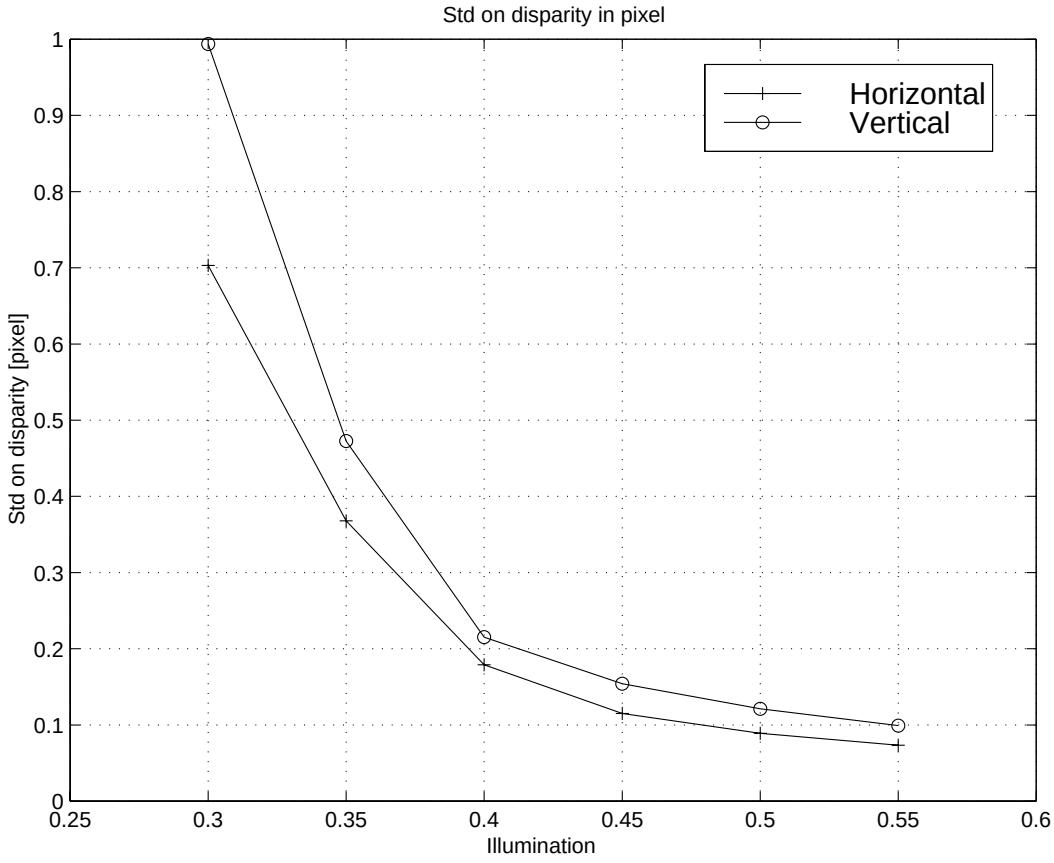


Figure 5.16: Standard deviation of displacement with respect to illumination conditions on a gravel scene.

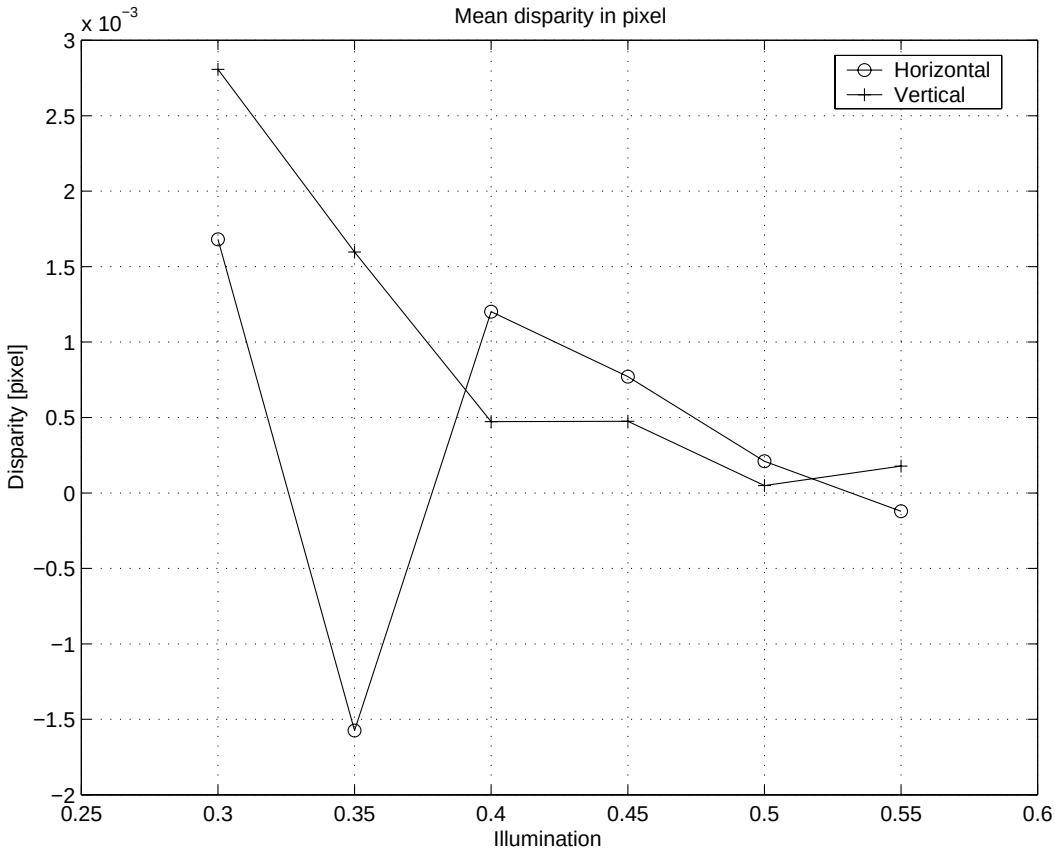


Figure 5.17: Average displacement measures with respect to illumination conditions on a gravel scene.

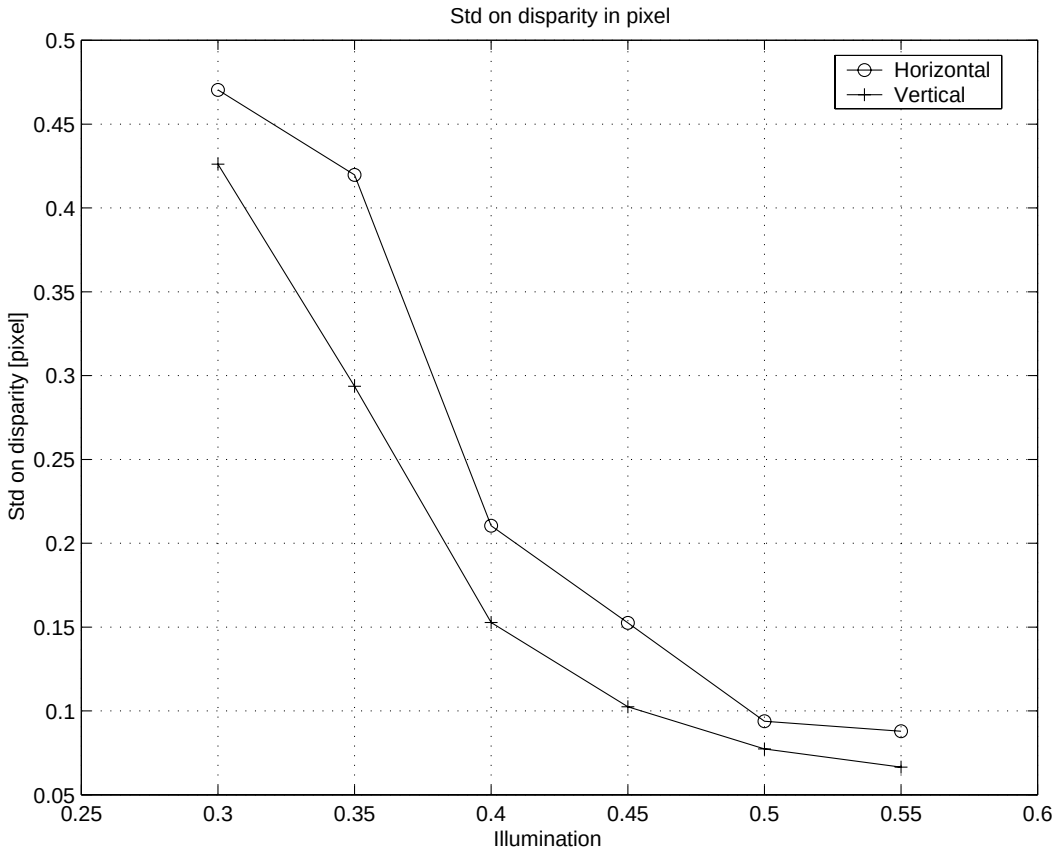


Figure 5.18: Standard deviation of displacement with respect to illumination conditions on a gravel scene.

5.3.6 Other limitations

During the course of these experiments, a small number of undesirable behaviours were noticed, and are mentioned here. Tracking consistency was affected by mismatches, if a given feature was not localised properly. The measure of contrast (eq. (5.7)) is an approximation of the autocorrelation function (see [73], appendix A), therefore the location of a given feature is not defined along the edges. This results in a drift of the estimated feature position in the image which does not fit the global motion model. Dynamic changes in illumination may also have the same adverse effect: the camera may not move, but the contrast properties might slightly change, and the tracked feature might be localised further away than its actual position.

Although undesirable, these behaviours are inherent to this feature extraction and tracking method which is based on local correlation measurements. The use of photometric normalisation (see [28] and references therein) should reduce the sensitivity to dynamic illumination variations.

5.4 Conclusions

This chapter's main contribution was to characterise the performance of the feature tracker. The main results have been listed below:

- The maximum admissible interframe displacement was estimated to be of 4 pixels;
- The tracker was able to track features consistently with a standard deviation of 0.03 pixel. This measurement characterises possible drift;
- In reasonable illumination conditions, the tracking remained consistent (0.5 pixel standard deviation). It was also noted that accuracy was a function of brightness, less light resulted in less accuracy. Besides, if the scene was overlit, the tracker could not work.

The investigation of the use of motion prediction schemes such as Kalman filtering [1] combined with regions or areas of interest [83] could both improve the computation

time (currently 200 ms on 512×512 8-bit images) and the feature positioning. Also, a multi-scale approach should prove to be more robust. Multi-scale approaches have been successfully applied to the computation of optical flow, see for example [63], and the review paper [2] for a comparison of optical flow techniques. Multi-scaling should allow coarse positioning when the relative motion of objects with respect to the camera is important, and a finer positioning when it is small, possibly improving the maximum admissible interframe displacement at the expense of positioning accuracy.

Chapter 6

Application of 2 1/2 D visual servoing to UUV dynamic positioning

This chapter[1] demonstrates a visual servoing method to dynamically position a UUV. The proposed method was applied to the 6 degrees-of-freedom model of ANGUS introduced in chapter 2. This method was designed to take into account the underactuation of ANGUS, and its robustness was tested against sea current disturbances, target plane orientation, and noise in the images. The simulations presented in this chapter did not include the feature tracker — previously evaluated in chapter 5 — within the control loop. This will be considered later in chapter 7.

The chapter introduces the hybrid visual servoing technique proposed by Malis *et al.* [50]. A method suitable for UUV visual servoing is then derived from it. The chapter closes by evaluating the robustness of the proposed method with respect to sea current disturbances, target orientation, and noise in the image.

6.1 The 2 1/2 D visual servoing approach

This section examines the details of the 2 1/2 D visual servoing approach of Malis *et al.* [50]. This approach was chosen as a starting point for it combined the advantages of both 2-D visual servoing and 3-D visual servoing without any of their drawbacks (see section 1.1.1). These advantages are recalled in the list below:

- the control was partially carried out in the image, ensuring that the target did not leave the field of view. This was not possible to achieve with a 3-D visual servoing task.

[1] The work presented in this chapter is a revised version of the paper [46].

- the control of the rotational degrees of freedom were decoupled from the translational ones. This allowed an easier design of the robot's control loops.[2]

- the convergence of the visual control was analytically proven to be the whole hemisphere in front of the target, even in the presence of camera calibration errors [50].

- finally, it did not require a model of the target, and was demonstrated to work on both planar and non-planar scenes.

The objective of visual servoing is to position a robot from the *current configuration* to the *desired* one. Let $\mathcal{F}$ be the current reference frame attached to the robot, and $\mathcal{F}^*$ the reference frame attached to the robot in its desired position[3]. It was seen in chapter 4 that from the homography of a plane it was possible to obtain motion information up to a scale factor. In particular, the rotation ${}^c\mathbf{R}_d$ between the current frame $\mathcal{F}$ and the desired frame $\mathcal{F}^*$ can be fully recovered. Consequently, a control law that decouples the rotational degrees-of-freedom from the translational ones can be designed. The techniques of Malis *et al.* relating the image information to the camera velocity screw $\boldsymbol{\nu}_s$ via a matrix will first be presented. Then their proposed robot control will be introduced.

6.1.1 Interaction matrix

Control of the camera's orientation

A natural way to control the camera's orientation is to use the estimated rotation matrix between $\mathcal{F}$ and $\mathcal{F}^*$: ${}^c\mathbf{R}_d$. The camera will reach its desired orientation when ${}^c\mathbf{R}_d = \mathbf{I}_3$. Let Ω be the angle of the transposed[4] aforementioned rotation matrix, and $\mathbf{l}$ the axis of this rotation, i.e. ${}^c\mathbf{R}_d^T$, (where $\|\mathbf{l}\| = 1$ and $0 \leq \Omega < 2\pi$). Malis showed that the time derivative of the product of the angle and the support vector of the rotation axis $\Omega\mathbf{l}$ was:

[2]This is obviously true only if the robot's degrees-of-freedom are decoupled.

[3]For station keeping purposes, the *desired* position is also the *initial* position.

[4]There was a typo in [50], it is actually the axis and angle of the rotation between $\mathcal{F}^*$ and $\mathcal{F}$ which were used, hence the transpose.

$$\frac{d(\Omega \mathbf{l})}{dt} = [\mathbf{0}_3 \, \mathbf{L}_\omega(\Omega, \mathbf{l})] \, \boldsymbol{\nu}_s \tag{6.1}$$

where $\mathbf{0}_3$ is the 3×3 zero matrix , and where $\mathbf{L}_\omega$ is:

$$\mathbf{L}_\omega(\Omega, \mathbf{l}) = \mathbf{I}_3 - \frac{\Omega}{2} [\mathbf{l}]_\times + \left(1 - \frac{\text{sinc}(\Omega)}{\text{sinc}^2(\frac{\Omega}{2})}\right), \tag{6.2}$$

and where the cardinal sine function is classically defined as:

$$\text{sinc}(\Omega) = \begin{cases} 1 & \text{if } \Omega = 2k\pi \, (k \in \mathbb{Z}), \\ \frac{\sin(\Omega)}{\Omega} & \text{otherwise.} \end{cases} \tag{6.3}$$

The derivation of the expression of matrix $\mathbf{L}_\omega$ can be found in E. Malis' PhD thesis [49]. The determinant of this matrix is:

$$\det(\mathbf{L}_\omega) = \frac{1}{\text{sinc}^2(\frac{\Omega}{2})}. \tag{6.4}$$

Therefore the matrix is singular only for angles $\Omega = 2k\pi \, (k \in \mathbb{Z}^*)$, i.e. outside the workspace. Note that eq. (6.1) clearly shows that the rotational d.o.f. of the camera are decoupled from the translational ones.

Control of the camera's translation

The purpose of this section is to recall [49] how a point's *extended coordinates* (as defined in eq. 6.5) can be used to determine a relationship between its time derivative and the camera velocity screw. Let $\mathbf{M} = [X, Y, Z]^T$ be a 3-D point in Euclidean space. The projection of $\mathbf{M}$ onto the image plane is $\tilde{\mathbf{m}}_n = [x, y, 1]^T = [\frac{X}{Z}, \frac{Y}{Z}, 1]^T$. The *extended image point* $\tilde{\mathbf{m}}_e$ is defined by:

$$\tilde{\mathbf{m}}_e = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} X/Z \\ Y/Z \\ \ln(Z) \end{bmatrix} \tag{6.5}$$

and $z = \ln(Z)$ is an added co-ordinate. The reader should note that, although the depth Z is unknown, the difference $z - z^* = \ln(Z/Z^*)$ can be obtained from an

homography. Indeed, if $\mathbf{M}$ belongs to a plane, Malis showed that the ratio Z/Z^* is given by:

$$\frac{Z}{Z^*} = \frac{\mathbf{n}^{*T}\,\tilde{\mathbf{m}}^*}{\mathbf{n}^T\,\tilde{\mathbf{m}}}\det(\mathbf{H}) \tag{6.6}$$

where $\mathbf{H}$ is the homography matrix between the desired and the current views.

If the extended image point $\tilde{\mathbf{m}}_e$ is differentiated with respect to time, the following expression is obtained:

$$\frac{d\tilde{\mathbf{m}}_e}{dt} = \begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{z}\end{bmatrix} = \begin{bmatrix}\frac{\dot{X}}{Z} - \frac{X}{Z}\frac{\dot{Z}}{Z}\\ \frac{\dot{Y}}{Z} - \frac{Y}{Z}\frac{\dot{Z}}{Z}\\ \frac{\dot{Z}}{Z}\end{bmatrix} = \frac{1}{Z}\begin{bmatrix}1 & 0 & -\frac{X}{Z}\\ 0 & 1 & -\frac{Y}{Z}\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\dot{X}\\ \dot{Y}\\ \dot{Z}\end{bmatrix} = -\mathbf{L}_v(\tilde{\mathbf{m}}_e)\,\dot{\mathbf{M}} \tag{6.7}$$

Note that $\mathbf{L}_v$ is singular for $Z = 0$ or $Z \to \infty$ since its determinant $\det(\mathbf{L}_v) = 1/Z^3$, which is again outside the considered workspace.

The time derivative of $\mathbf{M}$ expressed in the moving frame $\mathcal{F}$ is:

$$\dot{\mathbf{M}} = [\,-\mathbf{I}_3\ [\mathbf{M}]_\times\,]\,\boldsymbol{\nu}_s. \tag{6.8}$$

The combination of eq. (6.8) and (6.7) yield:

$$\frac{d\tilde{\mathbf{m}}_e}{dt} = [\,\mathbf{L}_v(\tilde{\mathbf{m}}_e)\ \mathbf{L}_{(v,\omega)}(\tilde{\mathbf{m}}_e)]\,\boldsymbol{\nu}_s \tag{6.9}$$

Now, since $\mathbf{M} = Z\,\tilde{\mathbf{m}}_n$, the expression of the matrix $\mathbf{L}_{(v,\omega)}(\tilde{\mathbf{m}}_e)$ is:

$$\mathbf{L}_{(v,\omega)}(\tilde{\mathbf{m}}_e) = Z\,\mathbf{L}_v(\tilde{\mathbf{m}}_e)\,[\tilde{\mathbf{m}}_n]_\times = \begin{bmatrix} xy & -(1+x^2) & y\\ (1+y^2) & -xy & -x\\ -y & x & 0\end{bmatrix} \tag{6.10}$$

Combining both rotation and translation

Now, if eq. (6.9) and (6.1) are combined, the following equation is obtained:

$$\frac{d}{dt}\begin{bmatrix}\tilde{\mathbf{m}}_e \\ \Omega\mathbf{l}\end{bmatrix} = \begin{bmatrix}\mathbf{L}_v & \mathbf{L}_{(v,\omega)} \\ \mathbf{0}_3 & \mathbf{L}_\omega\end{bmatrix}\begin{bmatrix}\boldsymbol{\nu}_{sp} \\ \boldsymbol{\nu}_{so}\end{bmatrix} = \mathbf{L}(\tilde{\mathbf{m}}_e)\,\boldsymbol{\nu}_s \tag{6.11}$$

the complete expression of the *interaction matrix* $\mathrm{d}\mathbf{L}(\tilde{\mathbf{m}}_e)$ is:

$$\mathbf{L}(\tilde{\mathbf{m}}_e) = \begin{bmatrix} -1/Z & 0 & x/Z & xy & -(1+x^2) & y \\ 0 & -1/Z & y/Z & 1+y^2 & -xy & -x \\ 0 & 0 & -1/Z & -y & x & 0 \\ 0 & 0 & 0 & 1-\beta(1+\mathbf{l}_x^2) & \beta\mathbf{l}_y\mathbf{l}_x - \alpha\mathbf{l}_z & \beta\mathbf{l}_z\mathbf{l}_x + \alpha\mathbf{l}_y \\ 0 & 0 & 0 & \beta\mathbf{l}_y\mathbf{l}_x - \alpha\mathbf{l}_z & 1-\beta(1+\mathbf{l}_y^2) & \beta\mathbf{l}_z\mathbf{l}_y - \alpha\mathbf{l}_x \\ 0 & 0 & 0 & \beta\mathbf{l}_z\mathbf{l}_x - \alpha\mathbf{l}_y & \beta\mathbf{l}_z\mathbf{l}_y + \alpha\mathbf{l}_x & 1-\beta(1+\mathbf{l}_z^2) \end{bmatrix} \tag{6.12}$$

where $\alpha = \frac{\Omega}{2}$ and $\beta = 1 - \frac{\mathrm{sinc}(\Omega)}{\mathrm{sinc}^2(\frac{\Omega}{2})}$. The only unknown in the interaction matrix $\mathbf{L}(\tilde{\mathbf{m}}_e)$ is the depth $Z = \rho\, d^*$, where ρ is given by:

$$\rho = \frac{\det(\mathbf{H})}{\mathbf{n}^T\,\tilde{\mathbf{m}}}. \tag{6.13}$$

Given the homography matrix $\mathbf{H}$ it is possible to estimate the normal vector $\mathbf{n}$ to the planar target. Hence the only real unknown of the interaction matrix is the distance of the camera to the plane in the desired configuration, that is d^*. Fortunately, Malis proved that the 2 1/2 D visual servoing scheme was robust to errors in the estimation of this distance [49, 50]. In practice, it is sufficient to estimate this distance in an off-line learning stage. The interaction matrix $\mathbf{L}(\tilde{\mathbf{m}}_e)$ is always of full rank, except if $Z = 0$, $Z \to \infty$ or $\Omega = 2k\pi\,(k \in \mathbb{Z}^*)$. Although the lower half of the matrix is fairly complex, by closing the loop, this part simplifies greatly, as will be seen in the next section.

6.1.2 A proportional control

As in [72], a robot's positioning task can be described as the regulation to zero of an error function: the *task function*. This function has to obey some properties

of regularity which are fully described in Samson's book [72]. The following task function, $\mathbf{e}$, obeys these conditions. Malis *et al.* defined $\mathbf{e}$ as:

$$\mathbf{e} = \begin{bmatrix} \tilde{\mathbf{m}}_e - \tilde{\mathbf{m}}_e^* \\ \Omega \mathbf{l} \end{bmatrix} \tag{6.14}$$

and $\tilde{\mathbf{m}}_e - \tilde{\mathbf{m}}_e^* = [x - x^*, y - y^*, \ln(\frac{Z}{Z^*})]^T$ can be computed from the current and the desired (or initial) images, and from ρ (see eq. (6.15)). x^* and y^* are the co-ordinates of the image of a target point in the desired image, and x and y are the co-ordinates of the same point in the current image. This point is the *controlled point*.

$$\frac{Z}{Z^*} = \rho \left(\mathbf{n}^{*T} \tilde{\mathbf{m}}^* \right) \tag{6.15}$$

Now, a relationship between the camera velocity $\boldsymbol{\nu}_s$ and the time derivative of the task function $\mathbf{e}$ will be established. If $\mathbf{W} = {}^s\mathbf{T}_v$[5] denotes the homogeneous transform[6] between the camera velocity $\boldsymbol{\nu}_s$ and the vehicle's $\boldsymbol{\nu}$, so that:

$$\boldsymbol{\nu}_s = \mathbf{W}\, \boldsymbol{\nu} \tag{6.16}$$

then:

$$\dot{\mathbf{e}} = \mathbf{L}\, \mathbf{W}\, \boldsymbol{\nu} \tag{6.17}$$

where $\mathbf{L}$ is the interaction matrix given by eq. (6.12). If the target is motionless, an exponential decrease of the task function vector on each of its components can be obtained by setting:

$$\dot{\mathbf{e}} = -\lambda\, \mathbf{K}_P\, \mathbf{e} \tag{6.18}$$

where λ is a positive scalar and $\mathbf{K}_P$ is a positive diagonal matrix which tunes the rate of convergence of $\mathbf{e}$ to zero. Combining (6.17) and (6.18) yields:

[5] The superscript s stands for sensor, and v stands for vehicle.

[6] See 4.1.1 for the definition of homogeneous transforms.

$$\boldsymbol{\nu} = -\lambda \mathbf{W}^{-1}\,\mathbf{L}^{-1}\,\mathbf{D}\,\mathbf{e} \tag{6.19}$$

Finally, expanding eq. (6.19), and choosing the matrix $\mathbf{K}_P$ as unitary:

$$\boldsymbol{\nu} = \begin{bmatrix} \boldsymbol{\nu}_p \\ \boldsymbol{\nu}_o \end{bmatrix} = -\lambda \begin{bmatrix} {}^s\mathbf{R}_v^T & -{}^s\mathbf{R}_v^T\,[{}^s\mathbf{t}_v]_\times \\ 0 & {}^s\mathbf{R}_v^T \end{bmatrix} \begin{bmatrix} \mathbf{L}_v{}^{-1} & -\mathbf{L}_v{}^{-1}\,\mathbf{L}_(v,\omega)\,\mathbf{L}_\omega{}^{-1} \\ 0 & \mathbf{L}_\omega{}^{-1} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{m}}_e - \tilde{\mathbf{m}}_e^* \\ \Omega\mathbf{l} \end{bmatrix} \tag{6.20}$$

Now, it was proven that [49]:

$$\mathbf{L}_\omega{}^{-1}\,\Omega\mathbf{l} = \left(\mathbf{I}_3 + \frac{\Omega}{2}\,\mathrm{sinc}^2(\frac{\Omega}{2})\,[\mathbf{l}]_\times + (1 - \mathrm{sinc}((\Omega)))\,[\mathbf{l}]_\times^2\right)\Omega\mathbf{l} = \Omega\,\mathbf{l} \tag{6.21}$$

The proportional control of the rotational d.o.f. of the vehicle can hence be written as $\boldsymbol{\nu}_o = -\lambda\,{}^s\mathbf{R}_v^T\,\Omega\mathbf{l}$, and $\mathbf{L}_\omega{}^{-1}$ can be set as $\mathbf{L}_\omega{}^{-1} = \mathbf{I}_3$.

In the following section, the modifications made to accommodate the original 2 1/2 D visual servoing method to the control particularities of ANGUS are detailed.

6.2 Extension of 2 1/2 D visual servoing to underwater vehicle station keeping

6.2.1 Introduction

The main objective to be achieved in this thesis was to dynamically position a UUV at a given start position in the presence of sea current disturbances. To simulate a UUV, the dynamic model of ANGUS presented in chapter 2 was used. Although in theory, it would be possible to apply the 2 1/2 D approach as proposed in [50], this was in fact impractical: doing so could lead the task function $\mathbf{e}$ to define a trajectory in 3-D space which would be physically unattainable by the vehicle. Indeed, the roll and pitch degrees-of-freedom being uncontrollable, any trajectory requiring specific roll and pitch values to be reached would fail. The best case scenario would be that these uncontrollable d.o.f. would act as perturbations. At worst, the vehicle could

reach a dynamic equilibrium (local minimum) and never converge to its desired position. The approach outlined in section 6.1, well-suited for a fully controllable robot, such as a 6-d.o.f. manipulator, was therefore not adapted to the specific control problem outlined in this thesis. A solution to this issue with the example of ANGUS will be presented now. Note that this could readily be extended to other types of underactuated[7] robots.

6.2.2 Modified control scheme

Since the underwater vehicle only had four controllable d.o.f., namely surge, sway, heave and heading, it was necessary to take this restriction into account in the visual servoing design. Failing to do so would lead to trajectories physically impossible to achieve by the robot. Let $\boldsymbol{\nu}_r = [u, v, w, r]^T$ be the vector of the controllable vehicle's velocities, that is surge, sway, heave and yaw rate. Let $\mathbf{J}_r$ be the 6×4 matrix that maps $\boldsymbol{\nu}_r$ onto $\boldsymbol{\nu}_s$:

$$\boldsymbol{\nu}_s = \mathbf{J}_r \, \boldsymbol{\nu}_r \tag{6.22}$$

In these simulation experiments, the camera reference frame was chosen to coincide with the vehicle's body reference frame. In other words, the camera possessed the dynamic characteristics of ANGUS. Therefore, there was a straight one to one relationship between the camera velocity components and the vehicle's ones as per eq. (6.16). Roll and pitch rates were not controllable. To eliminate them, the following expression of $\mathbf{J}_r$ was chosen:

$$\mathbf{J}_r = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.23}$$

[7]Here by underactuated, it is meant that not all the d.o.f. of the robot are controllable.

Now, the task function $\mathbf{e}$, being defined as in section 6.1, is related to the camera velocity screw by eq. (6.17). Hence, combining eq. (6.17) with eq. (6.22) yields:

$$\dot{\mathbf{e}} = \mathbf{L}\,\mathbf{J}_r\,\boldsymbol{\nu}_r = \mathbf{L}_r\,\boldsymbol{\nu}_r \tag{6.24}$$

¿From eq. (6.24) and (6.12), the expression of the *reduced order interaction matrix* $\mathbf{L}_r$ can be obtained:

$$\mathbf{L}_r = \begin{bmatrix} -1/Z & 0 & x/Z & 0 & 0 & y \\ 0 & -1/Z & y/Z & 0 & 0 & -x \\ 0 & 0 & -1/Z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \beta \mathbf{l}_z \mathbf{l}_x + \alpha \mathbf{l}_y \\ 0 & 0 & 0 & 0 & 0 & \beta \mathbf{l}_z \mathbf{l}_y - \alpha \mathbf{l}_x \\ 0 & 0 & 0 & 0 & 0 & 1 - \beta(1 + \mathbf{l}_z^2) \end{bmatrix} \tag{6.25}$$

Achieving an exponential decrease of the task function $\mathbf{e}$ can be obtained by choosing a positive scalar λ so that $\dot{\mathbf{e}} = -\lambda\,\mathbf{e}$. λ is controlling the speed of convergence. This choice would be (and is sufficient) for a conventional positioning task. However, in an underwater environment, an UUV is likely to be subject to external disturbances such as sea currents or, if tethered (ROV), subject to cable disturbances. For example, the case where ANGUS was subject to a constant step input in sea current velocity was studied (see 2.2.5, and 2.3 eq. (2.31)). With a mere proportional control, the closed loop system: visual controller plus ANGUS model is of type 0. Therefore, a steady-state error in position would be present. It is then necessary to add some integrators within the controller to obtain a type 1 system [17], hence no steady-state error in position.

In order to offset constant sea current disturbances, a vectorial PID control was included, so that the vehicle's desired speed was:

$$\boldsymbol{\nu}_r = -\lambda\,\mathbf{L}_r^{+}\left(\mathbf{K}_P\,\mathbf{e} + \mathbf{K}_D\,\dot{\mathbf{e}} + \mathbf{K}_I \int_0^t \mathbf{e}\,dt\right) \tag{6.26}$$

where $\mathbf{K}_P$ (6×6), $\mathbf{K}_D$ (6×6), and $\mathbf{K}_I$ (6×6) are positive diagonal matrices. $\mathbf{L}_r^{+}$ is the pseudo-inverse of $\mathbf{L}_r$. Since $\mathbf{L}_r$ is a rank 4 matrix, its pseudo-inverse is given by:

$$\mathbf{L}_r^+ = (\mathbf{L}_r^T \mathbf{L}_r)^{-1} \mathbf{L}_r^T \tag{6.27}$$

Once the desired speeds are computed, a simple thruster mapping with saturation terms is applied as shown on Fig. 6.1.

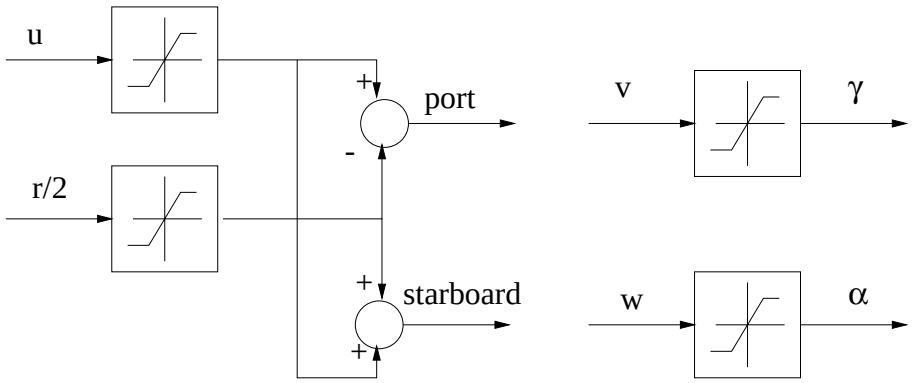


Figure 6.1: Thruster mapping and saturation blocks of ANGUS 003.

Simulation results of this technique are given in the following section.

6.3 Performance evaluation of the modified 2 1/2 D visual servoing

The previous section described the theory of the proposed visual servoing scheme. The objective of this section is to evaluate the performance of this control scheme under various conditions.

"Nominal" station-keeping tests were performed. A planar target was placed parallel to the camera focal plane while ANGUS' simulation was subject to constant sea current disturbances.

The effect of the relative orientation between the planar target and the focal plane on the visual servoing behaviour was studied.

Finally, the position accuracy obtainable if the feature tracking was subjected to noise was characterised.

6.3.1 Common setup of the simulation experiments

The vehicle was initially motionless and placed at the origin of the world reference frame, the initial state was: $\boldsymbol{\eta}_{(t=0)} = [0, 0, 0, 0, 0, 0]^T$. At time $t = 0$, a step input disturbance in sea current was applied to the vehicle's simulation. The current was characterised by its velocity vector, expressed in the world reference frame: $\mathbf{V}_c = [\dot{X}_c, \dot{Y}_c, \dot{Z}_c]^T$. The vehicle's ability to come back to its initial position in stable equilibrium was then studied. In these simulations, the following assumptions were made:

- The camera was a pinhole model as described in 4.1.3. The intrinsic parameters used were the ones extracted from the underwater camera examined in chapter 5 (table 5.1).
- The feature matching was perfect. Tracking was also perfect in the first set of tests, then noise was added to the tracking process in the other tests.
- The target scene was planar and comprised five points of which no three were collinear so that the homography matrix would not be singular. The 3-D points were placed 3 metres away from the focal plane of the camera.

It was also assumed that the camera was moving with the dynamics of ANGUS; the co-ordinate transformation from the camera to the vehicle ${}^s\mathbf{T}_v$ was thus constant and equal to:

$$ {}^s\mathbf{T}_v = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3^T & 1 \end{bmatrix}. \tag{6.28} $$

The chosen planar target was made up of five points belonging to the plane of equation $Z = 3$, 3 metres away from the camera focal plane at time $t = 0$:

$$
\begin{aligned}
\mathbf{M}_1 &= [0.25, 0.25, 3]^T \\
\mathbf{M}_2 &= [0.25, -0.25, 3]^T \\
\mathbf{M}_3 &= [-0.25, -0.25, 3]^T \\
\mathbf{M}_4 &= [-0.25, 0.25, 3]^T \\
\mathbf{M}_5 &= [0.5, 0.1, 3]^T
\end{aligned}
$$

The plane was oriented so that its normal vector $\mathbf{n}(t = 0) = \mathbf{n}^* = [0, 0, 1]^T$, was parallel to the Z-axis of ANGUS. Besides, in that configuration, the initial distance of the target plane to the vehicle was $d^* = 3\,\mathrm{m}$.

Throughout all the experiments the control parameters of the PID remained constant:

$$
\begin{aligned}
\mathbf{K}_P &= \mathrm{diag}(6.0, 4.0, 1.2, 0.0, 0.0, 0.5) \\
\mathbf{K}_D &= \mathrm{diag}(6.0, 6.0, 1.2, 0.0, 0.0, 3.0) \\
\mathbf{K}_I &= \mathrm{diag}(0.1, 0.2, 0.02, 0.0, 0.0, 0.01),
\end{aligned}
$$

and the sampling period T_s was set to 200 ms. This rather slow control rate (5 Hz) was chosen to be compatible with the real-time version of the feature tracker and of the visual servoing on the Cartesian robot. It also proved sufficient to ensure control stability (see chapter 7).

6.3.2 Influence of sea current disturbances

Protocol

The first set of simulations performed was aimed at assessing the validity and the performance of the modified control scheme to reject sea current disturbances, and dynamically position ANGUS. For that purpose, 176 runs with different values of sea current were carried out. The sea current values were obtained by combining:

- 11 values for $\dot{X}_c$: from 0 m/s to -0.9 m/s in steps of -0.1 m/s,
- 4 values for $\dot{Y}_c$: from 0 m/s to -0.3 m/s in steps of -0.1 m/s,
- and 4 values for $\dot{Z}_c$: from 0 m/s to -0.3 m/s in steps of -0.1 m/s.

All the above runs were successful, i.e. after a transition phase, the robot came back to its initial and desired position without positioning error and remained stable. As the magnitude of the sea current increased, the time response increased accordingly. A run was not considered to fail when the thruster power available was not sufficient to counteract the action of the sea current. That was the case for sea current velocities so that $\dot{X}_c < -1$ m/s , or $\dot{Y}_c < -0.3$ m/s , or $\dot{Z}_c < -0.3$ m/s. Note that all sea current velocity components were negative, because ANGUS' thrusters were more efficient when operating in forward mode rather than reverse.

Results

For illustrative purposes, three typical experiments with increasing amplitudes of sea current velocities have been selected. Note that ANGUS' capabilities to counteract the action of the current were not exceeded.

- **Run 1: weak sea current**

  For the first simulation, the sea current's velocity was set to $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s. The results are shown in figures 6.2, 6.3, 6.4, 6.5, 6.6, 6.7 and 6.8.

  The figures are organised as follows:

  - Figure 6.2 is the time history of the robot's position error $(\boldsymbol{\eta}_p(t) - \boldsymbol{\eta}_p(0))$, expressed in metres, with respect to its starting position at $t = 0$.
  - Figure 6.3 shows the robot's orientation errors in degree $(\boldsymbol{\eta}_o(t) - \boldsymbol{\eta}_o(0))$.
  - Figures 6.4 and 6.5 illustrate the evolution of the task function.
  - Figure 6.6 gathers the actual thruster values output of the PID controller.

- Figure 6.7 shows the evolution of the error in pixel co-ordinates, that is the difference between the desired horizontal (resp. vertical) co-ordinate of image point i: u_i^* (resp. v_i^*) and its current co-ordinate at time t: u_i (resp. v_i), $i \in [1, 5]$.
- Figure 6.8 represents the feature points' trajectories in the 512×512 image.

- **Run 2: moderate sea current**

  For the second run, the sea current was set to $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s. The results are similarly shown in figures 6.9, 6.10, 6.11, 6.12, 6.13, 6.14 and 6.15.

- **Run 3: strong sea current**

  For the third run, the sea current was set to $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s. The results are shown in figures 6.16, 6.17, 6.18, 6.19, 6.20, 6.21 and 6.22.

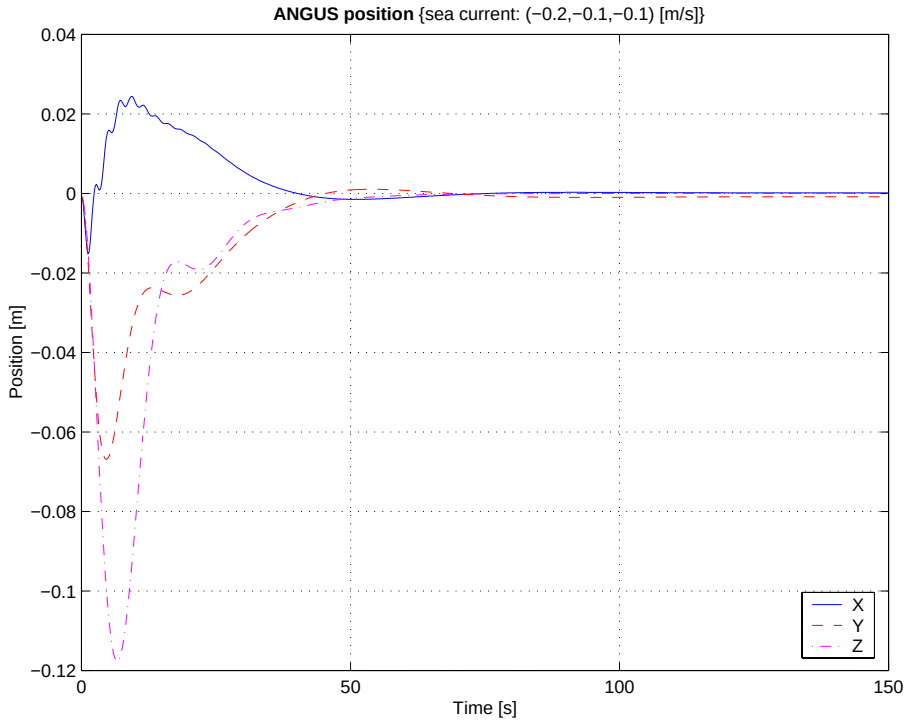


Figure 6.2: Run 1: weak sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s: position errors in metres.

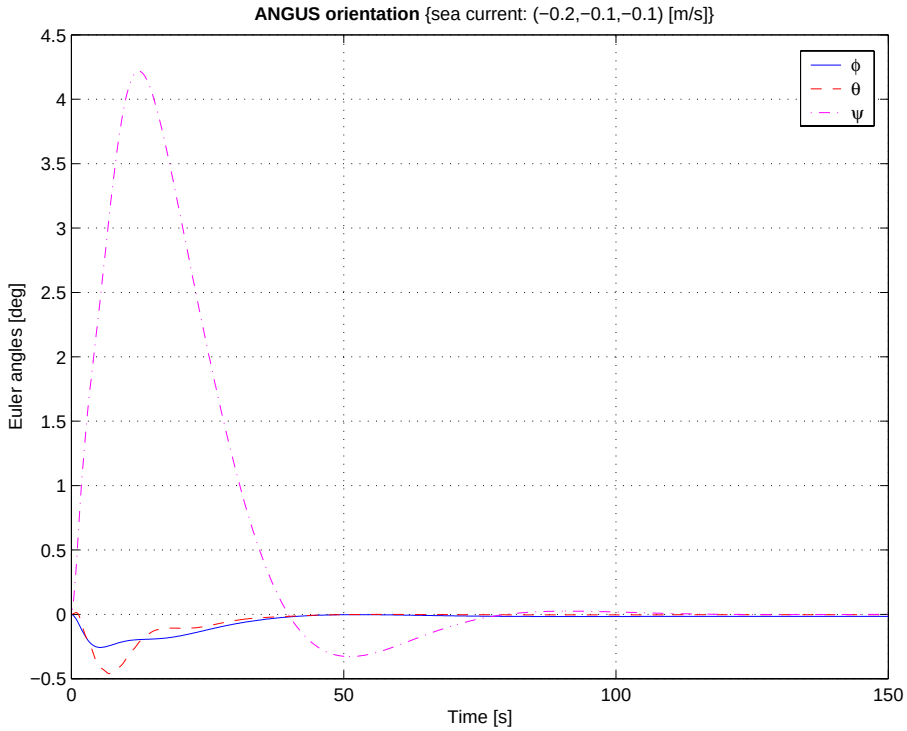


Figure 6.3: Run 1: weak sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s: orientation errors in degrees.

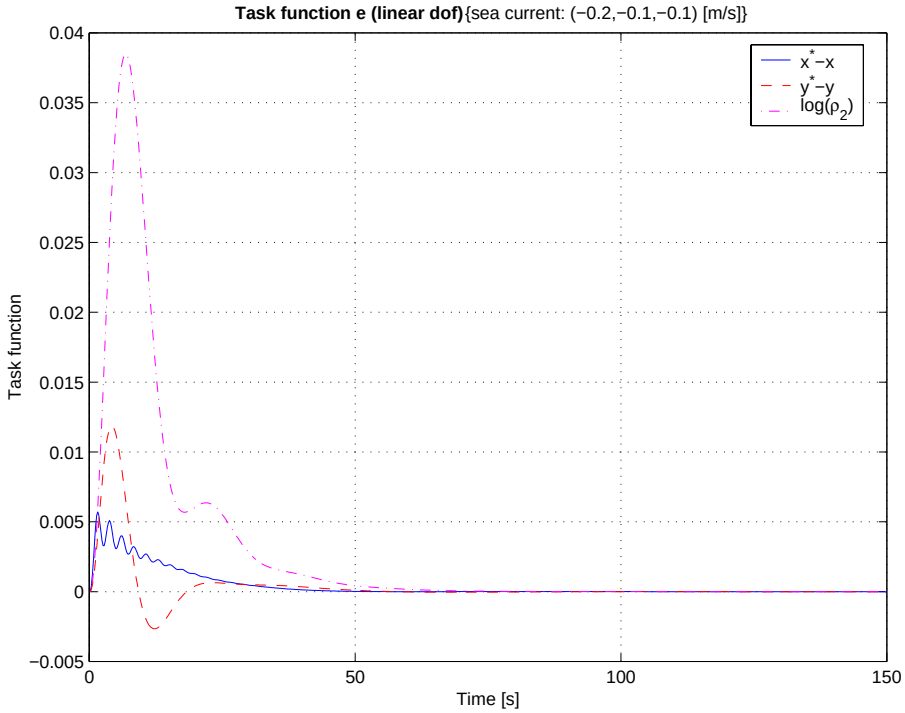


Figure 6.4: Run 1: weak sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s: translational components of the task function $\mathbf{e}$.

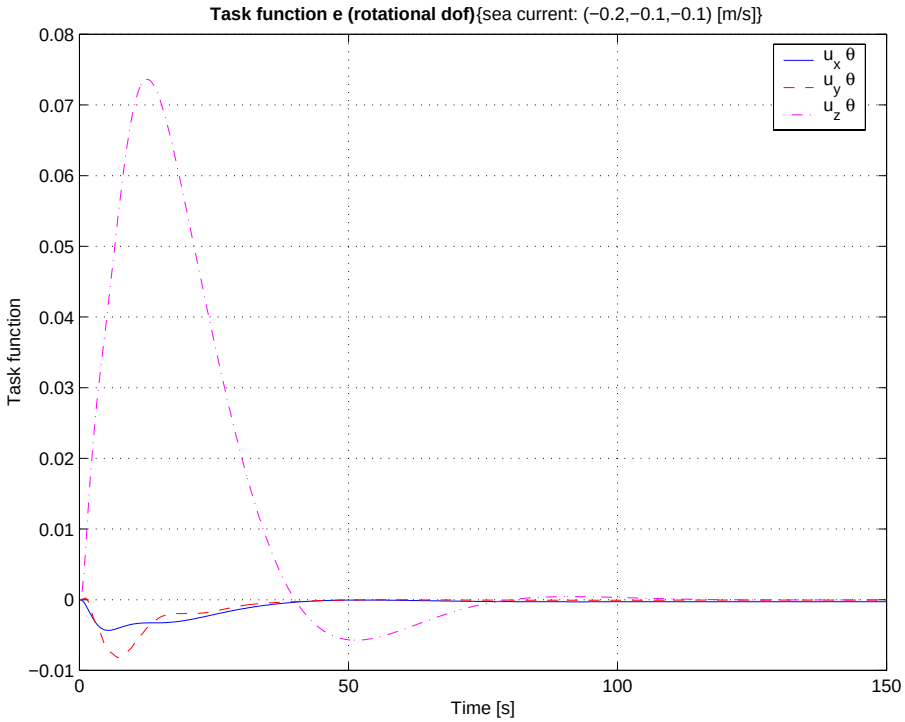


Figure 6.5: Run 1: weak sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s: rotational components of the task function $\mathbf{e}$.

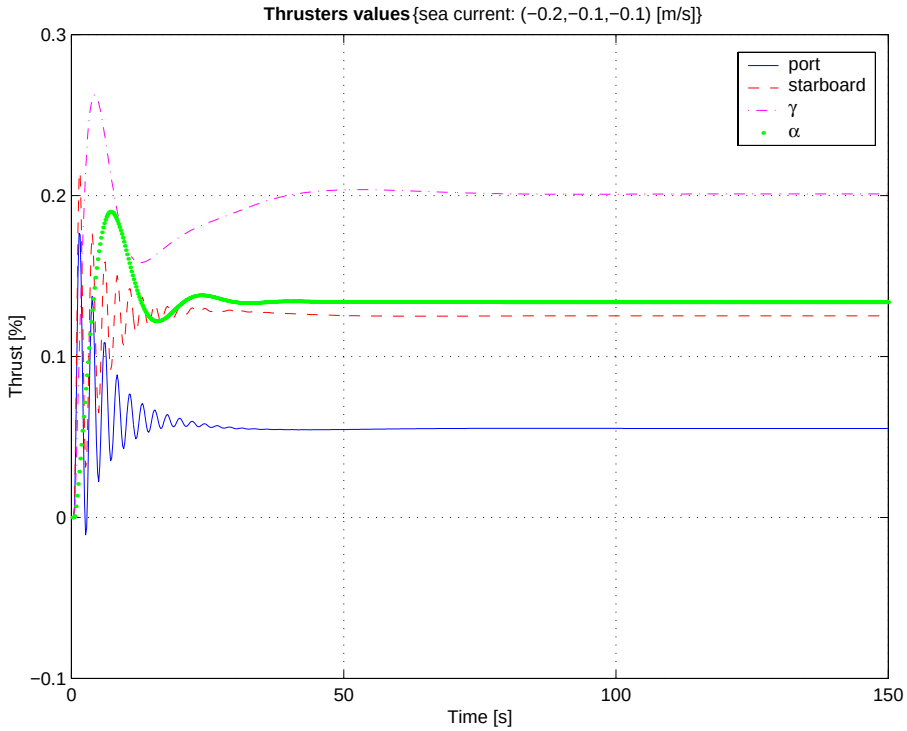


Figure 6.6: Run 1: weak sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s: thruster values.

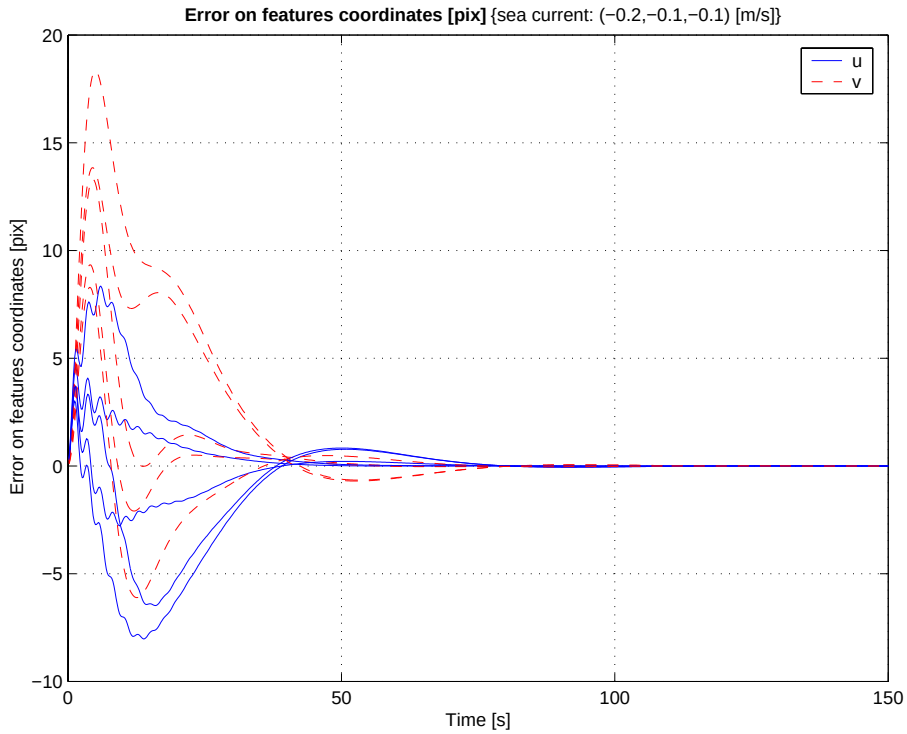


Figure 6.7: Run 1: weak sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s: errors in pixel co-ordinates.

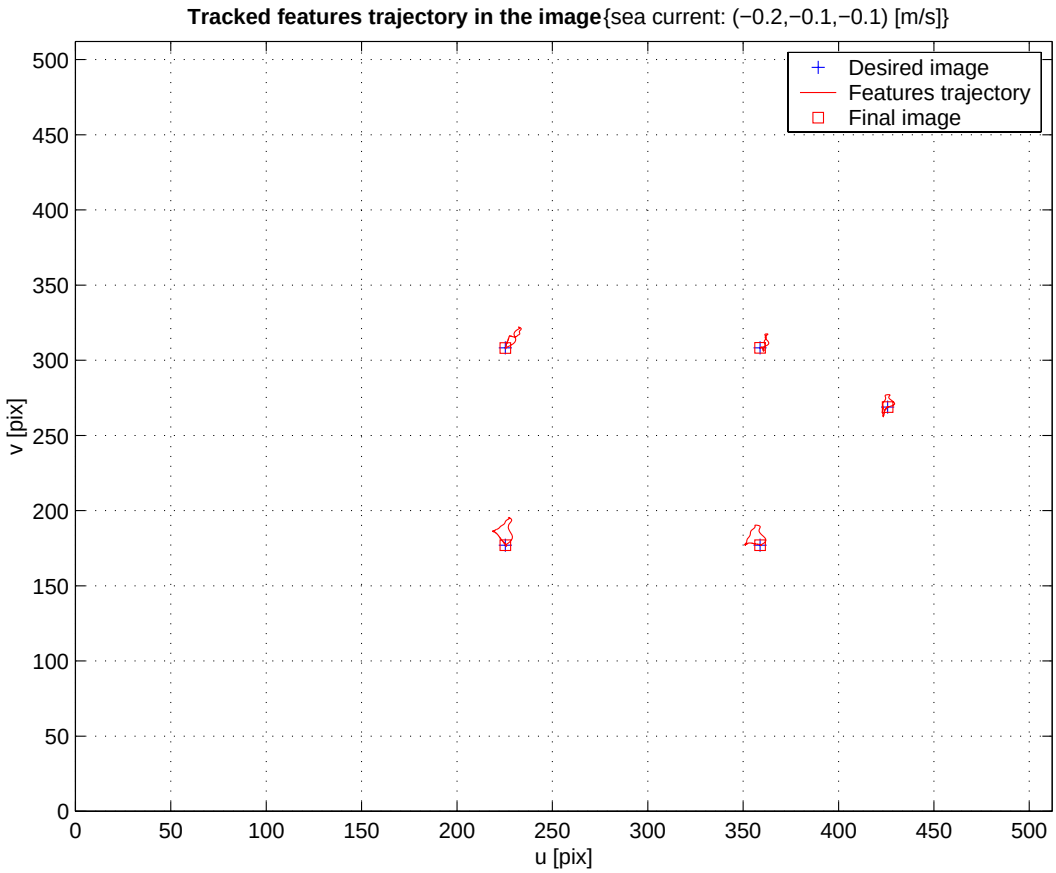


Figure 6.8: Run 1: weak sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s: trajectory of feature points in the image.

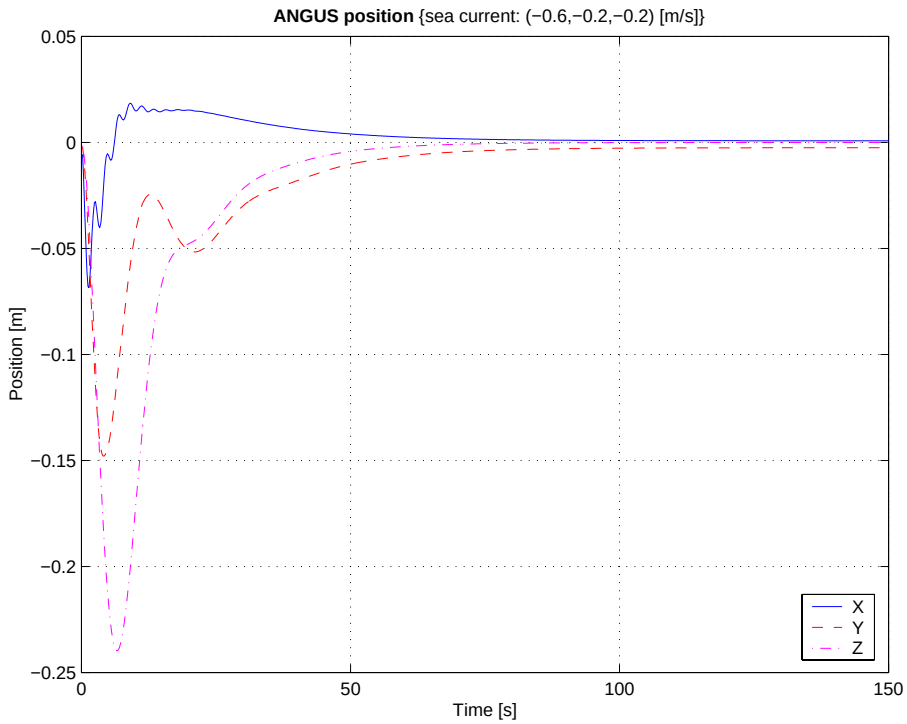


Figure 6.9: Run 2: moderate sea current $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s: position errors in metres.

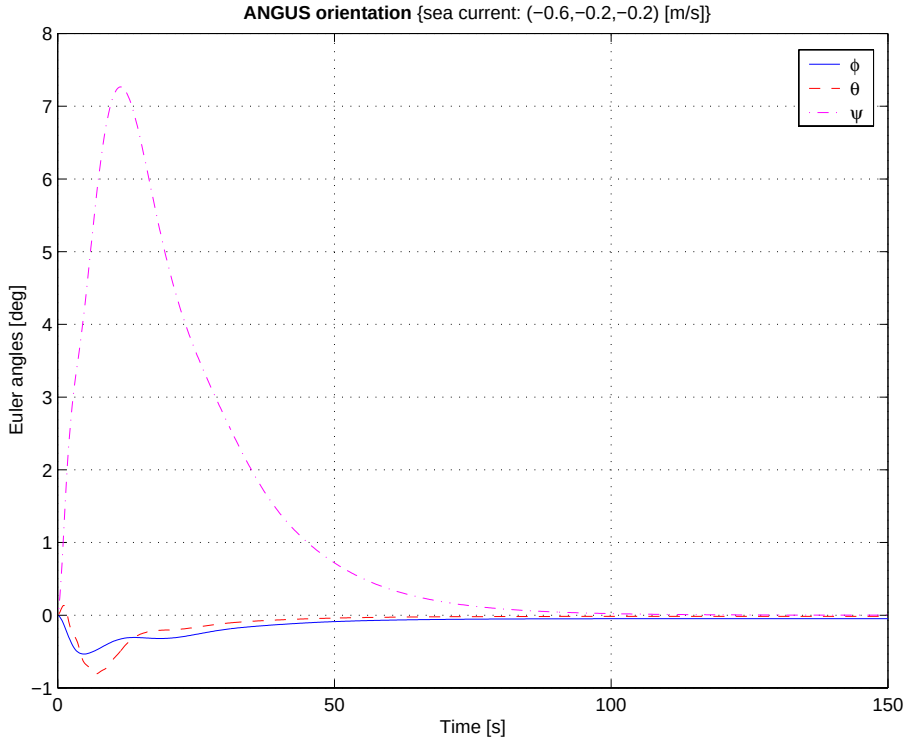


Figure 6.10: Run 2: moderate sea current $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s: orientation errors in degrees.

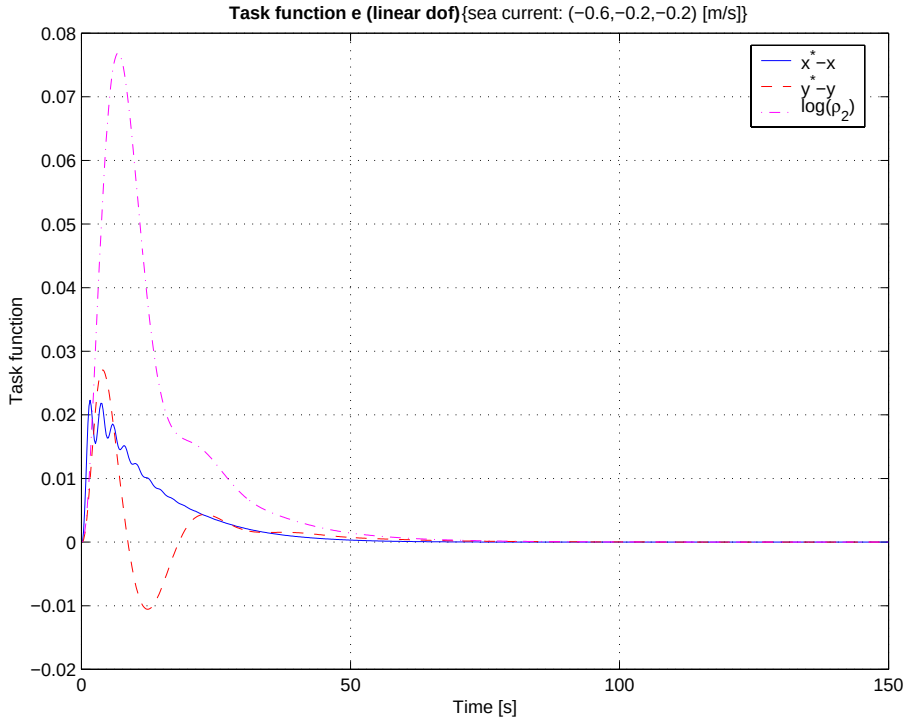


Figure 6.11: Run 2: moderate sea current $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s: translational components of the task function $\mathbf{e}$.

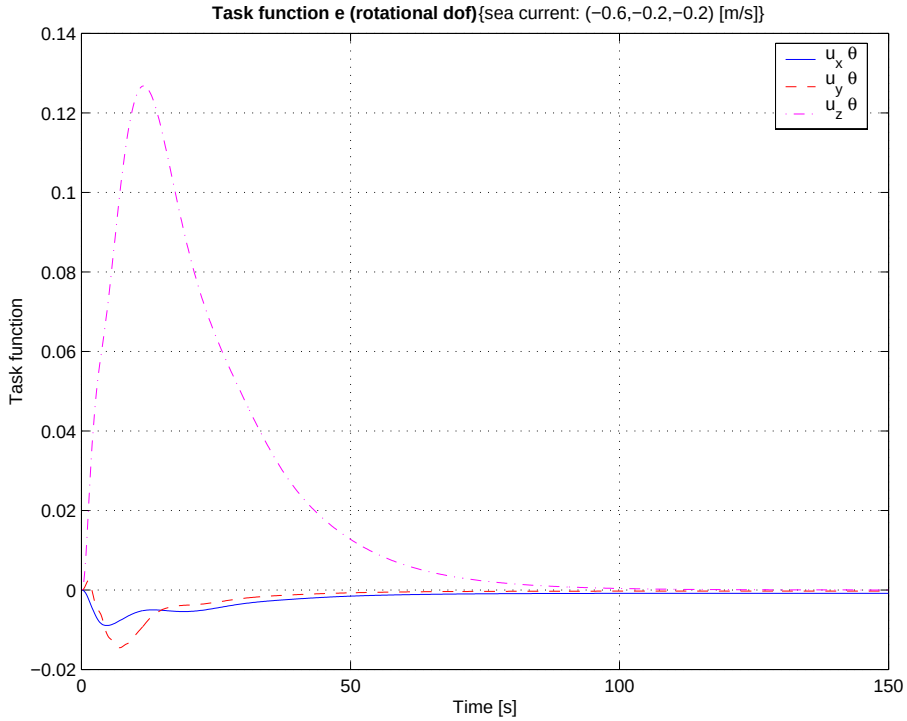


Figure 6.12: Run 2: moderate sea current $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s: orientational components of the task function $\mathbf{e}$.

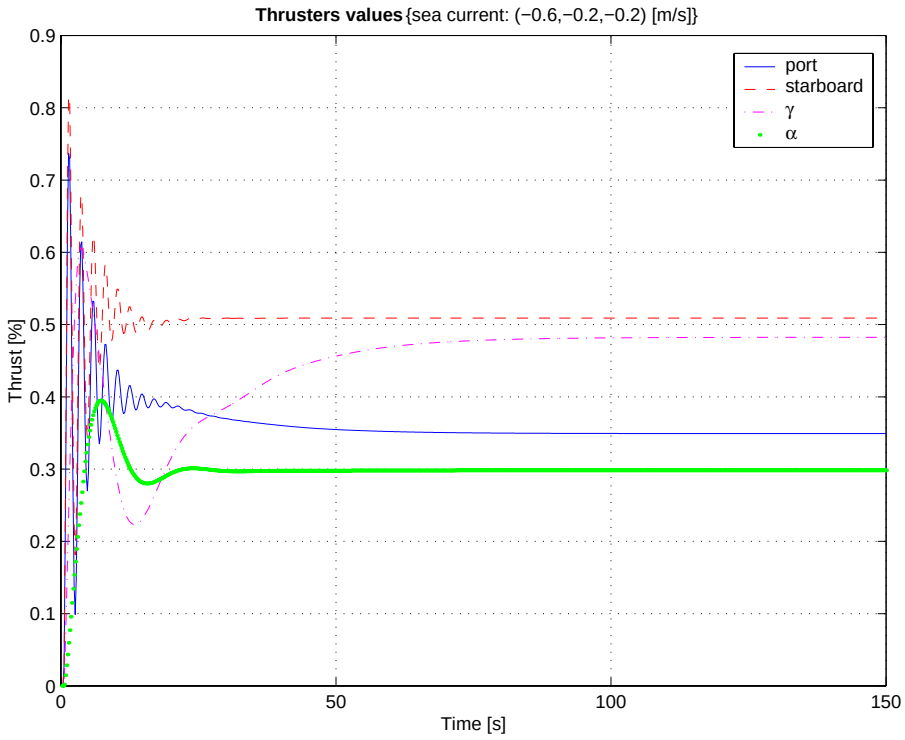


Figure 6.13: Run 2: moderate sea current $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s: thruster values.

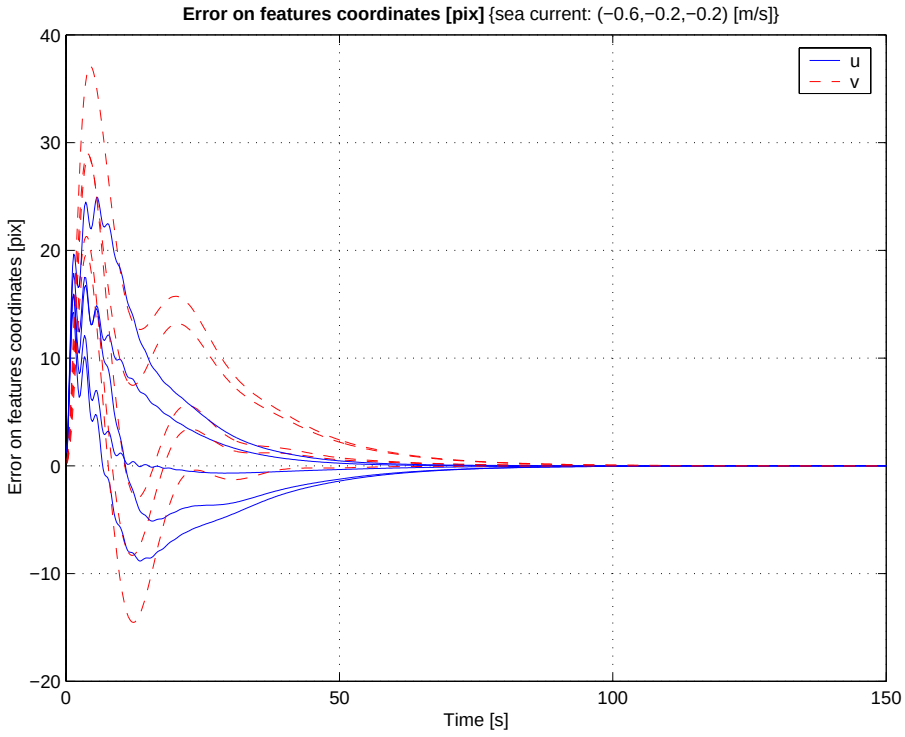


Figure 6.14: Run 2: moderate sea current $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s: errors in pixel co-ordinates.

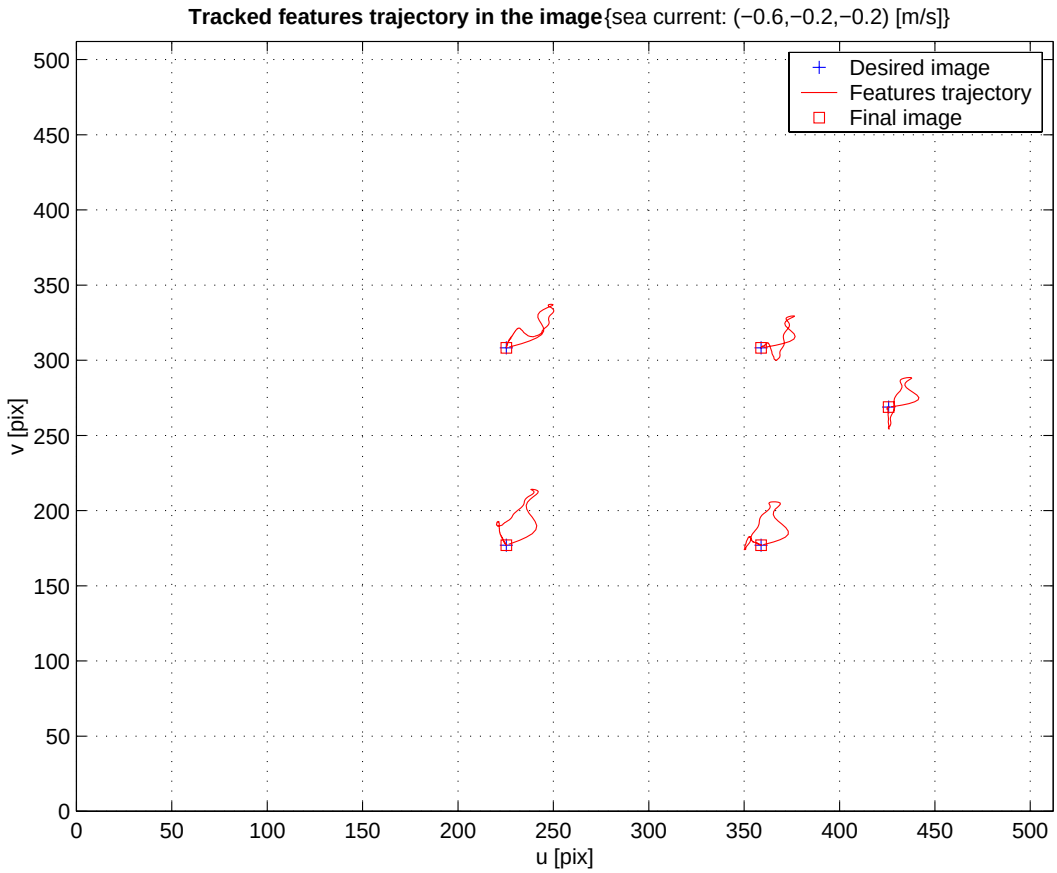


Figure 6.15: Run 2: moderate sea current $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s: trajectory of feature points in the image.

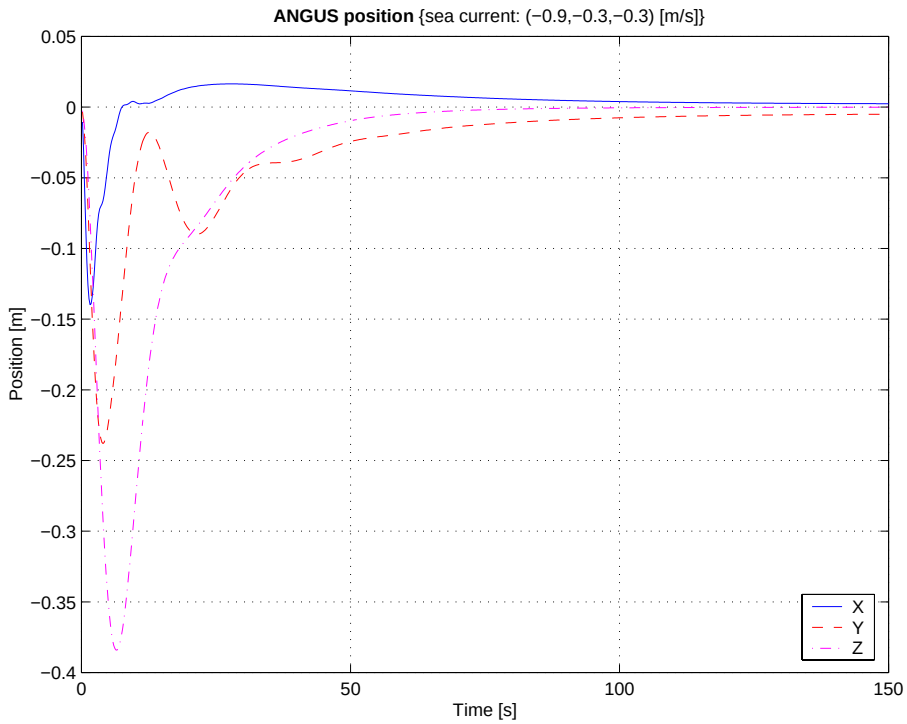


Figure 6.16: Run 3: strong sea current $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s: position errors in metres.

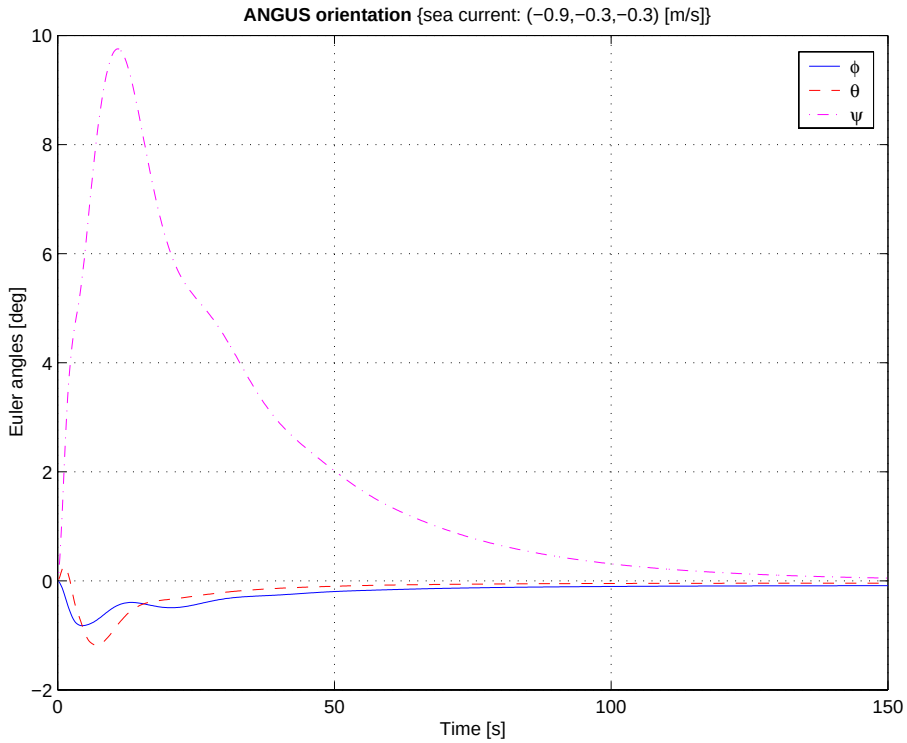


Figure 6.17: Run 3: strong sea current $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s: orientation errors in degrees.

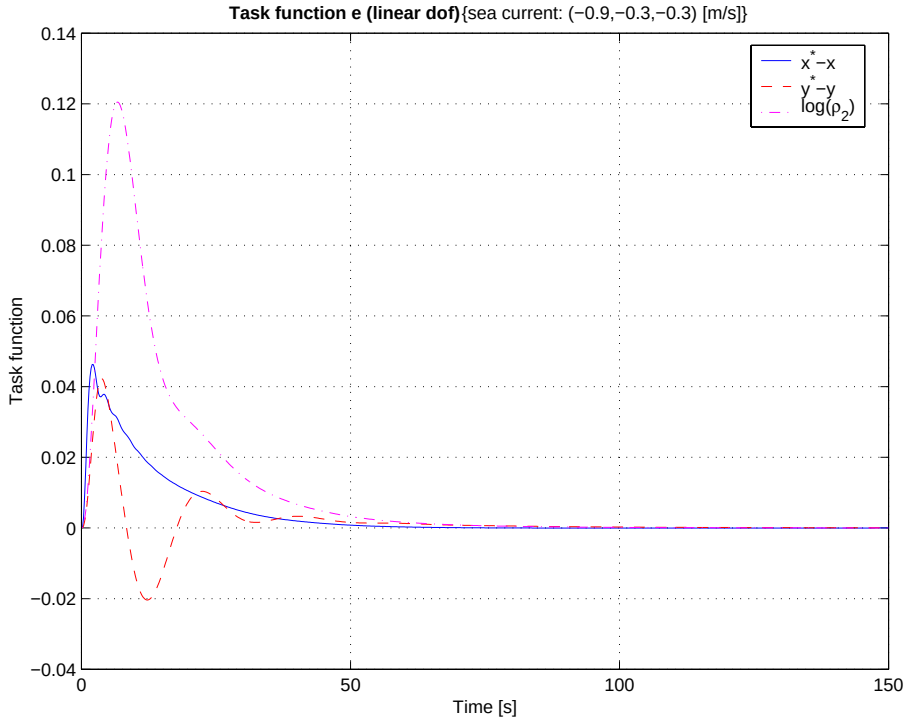


Figure 6.18: Run 3: strong sea current $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s: translational components of the task function $\mathbf{e}$.

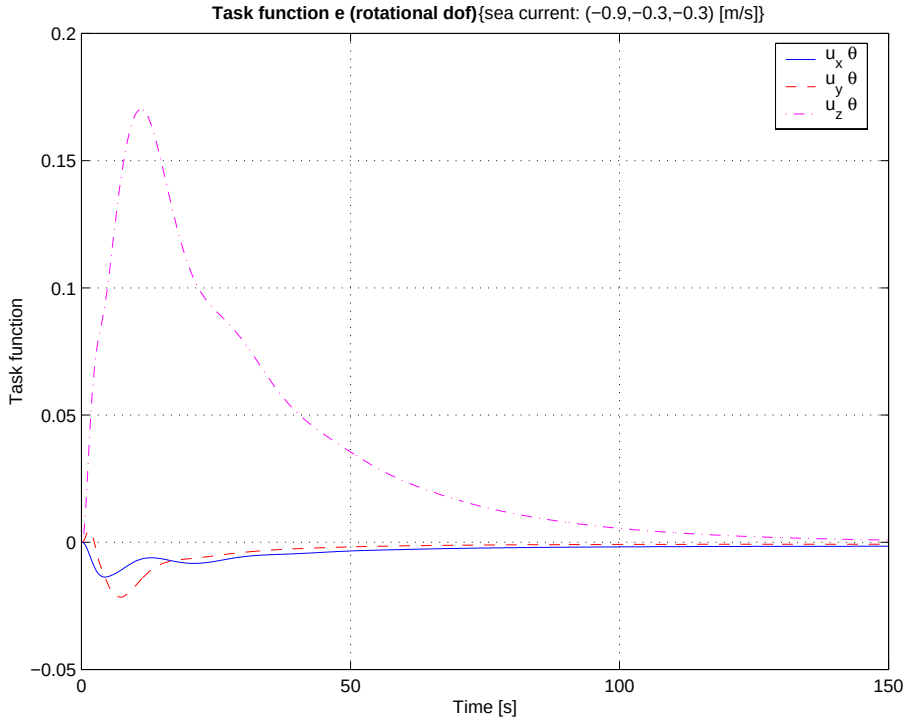


Figure 6.19: Run 3: strong sea current $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s: rotational components of the task function $\mathbf{e}$.

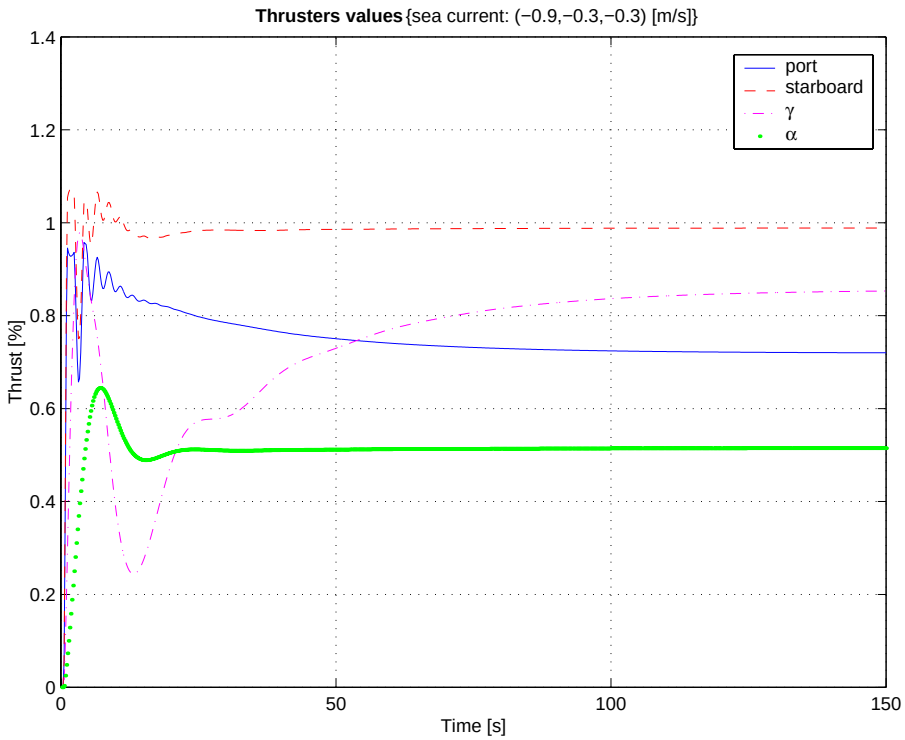


Figure 6.20: Run 3: strong sea current $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s: thruster values.

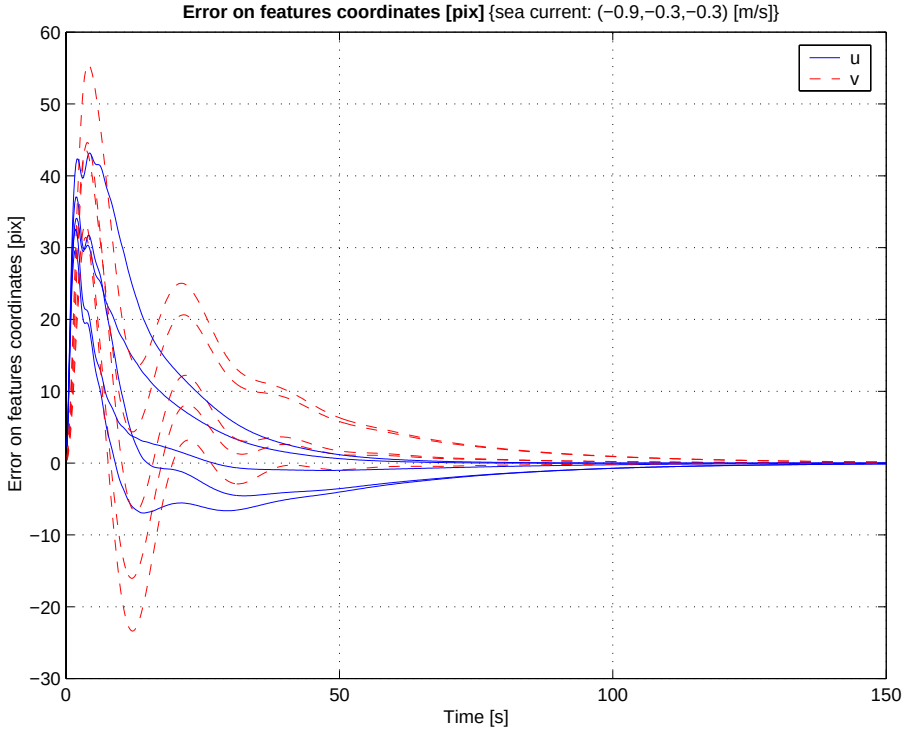


Figure 6.21: Run 3: strong sea current $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s: errors in pixel co-ordinates.

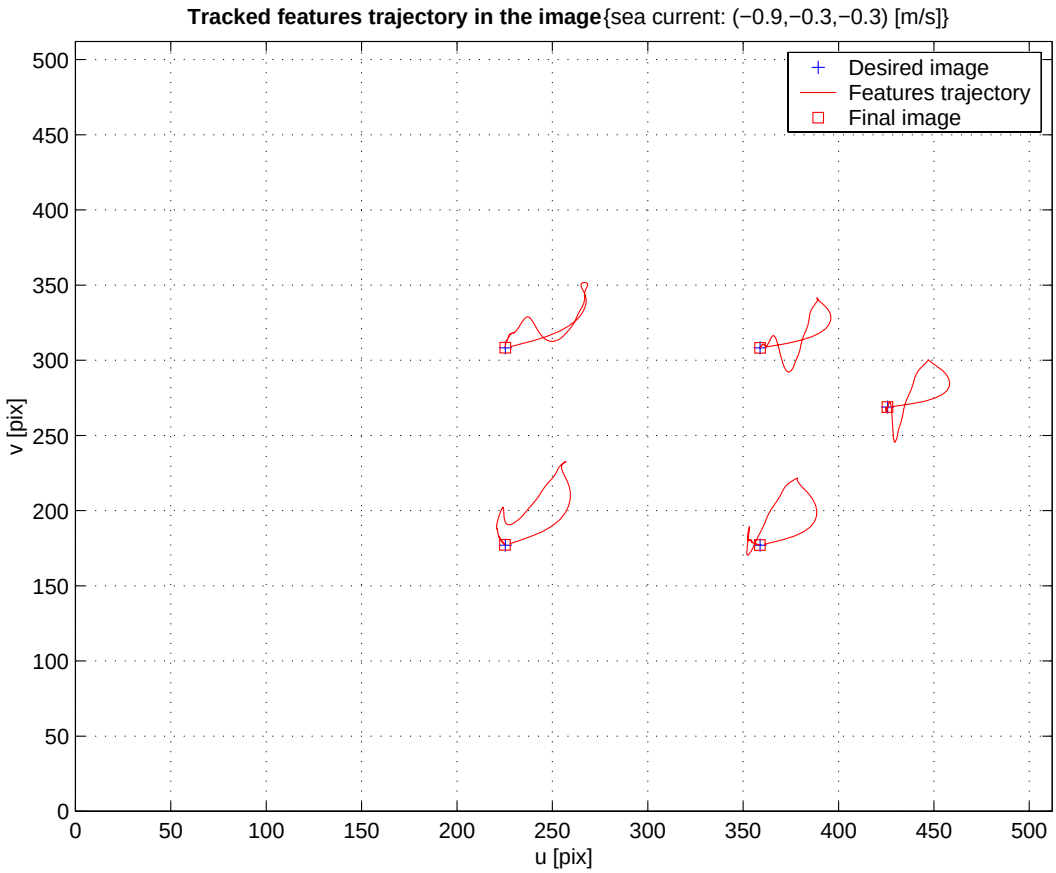


Figure 6.22: Run 3: strong sea current $\mathbf{V}_c = [-0.9, -0.3, -0.3]^T$ m/s: trajectory of feature points in the image

Discussion

In all three cases, after the initial displacement induced by the sea current's disturbance, the vehicle repositioned itself to within a close vicinity of the starting point. The dynamic positioning capability was therefore demonstrated in the presence of constant sea currents.

As expected, as the sea current's amplitude increased, the vehicle's response became slower. Indeed, the settling time varied from 50 s in the first illustrative run (weak current) up to 150 s in the third run (strong sea current). In fact, the greater the disturbance was, the more energy[8] the vehicle had to provide to regain its equilibrium point, and since its thruster power was limited, it took longer to come back to the initial position.[9]

However, close examination of the positioning errors, run 2 for instance (see figures 6.9 and 6.10), shows that they did not exactly reach zero. There are small static errors along and about the X and Y axes. In the case of run 2, the positioning errors are roughly of 0.8 mm along X and -2.5 mm along Y, while the angular errors are around -0.05 $^\circ$ in roll and -0.02 $^\circ$ in pitch. In other words, the vehicle reached a local minimum. This is due to two facts. Firstly, the roll and pitch degrees-of-freedom of ANGUS are not controllable. Secondly, angular motions and translation motions of the camera are coupled at the image level. A rotation of the camera about its X-axis (roll angle) induces, in the image plane, a translation of the imaged controlled target's point along the horizontal axis.[10] Conversely, a rotation of the camera about its Y-axis (pitch angle) produces a translation of the controlled target's point along the vertical axis of the image. Therefore, it is possible for the vehicle to reach an equilibrium point where the first two elements of the task function $\mathbf{e}$ (corresponding to the control of the X and Y position of the vehicle) reach a zero value, while the rotational components corresponding to roll and pitch angles (i.e. $\mathbf{e}_4$ and $\mathbf{e}_5$) are different from zero.[11] That was exactly the phenomenon observed in figures 6.11

[8] Or more exactly *work* in the physical sense.

[9] Since it is well-known that work = power $\times$ time.

[10] The *controlled point* was previously defined in section 6.1.2.

[11] Recall that $\mathbf{e}$ was defined in section 6.1 as a 6×1 vector by the equation:

$$\mathbf{e} = [\tilde{\mathbf{m}}_e - \tilde{\mathbf{m}}_e^*, \Omega\mathbf{l}]^T. \tag{6.29}$$

and 6.12.

Finally, noticeable oscillations in the port and starboard thrusters' curves of all runs may suggest some instability. In fact, the coefficients of the PID controller for the surge d.o.f. were deliberately set high. The main reasons behind this design choice were:

1. To minimise the deviation from the starting position of ANGUS to the furthest point of its trajectory,

2. To obtain better time response in sea current disturbance rejection.

As an example, reconsider run 1. With PID controller's gains set at lower values:

$$\begin{aligned} \mathbf{K}_P &= \text{diag}(\mathbf{0.1}, 4.0, 1.2, 0.0, 0.0, 0.5) \\ \mathbf{K}_D &= \text{diag}(\mathbf{0.0}, 6.0, 1.2, 0.0, 0.0, 3.0) \\ \mathbf{K}_I &= \text{diag}(\mathbf{0.001}, 0.2, 0.02, 0.0, 0.0, 0.01), \end{aligned}$$

(this set of coefficients will be referred to as the "slow PID setting"), the port and starboard thruster fast oscillations disappear. The port and starboard thrusters controlled both surge and heading by means of the thruster mapping illustrated in figure 6.1. It is then legitimate to wonder whether the oscillations originated from the surge control only, and not from the heading control. Since the heading plots of runs 1, 2 and 3 did not show any oscillations (see figure 6.3), it was clear that the oscillations originated from the surge control only. To illustrate the removal of the thrusters' ringing phenomenon, compare the evolution through time of the forward thrust $\boldsymbol{\nu}_{r1} = (port + stb)/2$, when the sea current was $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s, for both the "slow PID" set of coefficients, and the "fast PID" set of coefficients. This is illustrated by figure 6.23.

Note as well that a direct consequence of "detuning" the PID is to slow down the time response of the closed-loop system. Indeed, figure 6.24 clearly shows that the settling time of X increased from 75 s for the "fast PID setting" up to 160 s. Besides,

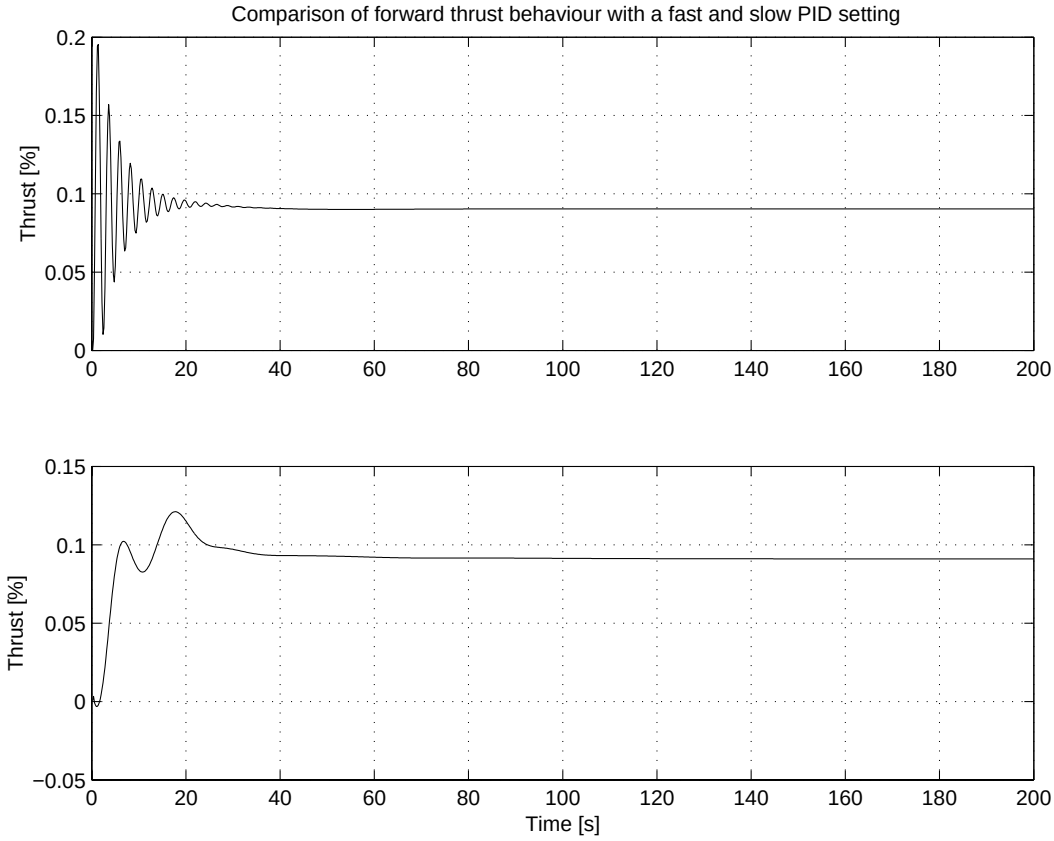


Figure 6.23: Comparison of forward thrust $\boldsymbol{\nu}_{r1} = (port + stb)/2$ through time when ANGUS is subject to a constant sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s with a "fast" (above), and "slow" (below) set of surge PID coefficients. Note that the "slow" PID did not show any ringing phenomenon on the thrusters' signal.

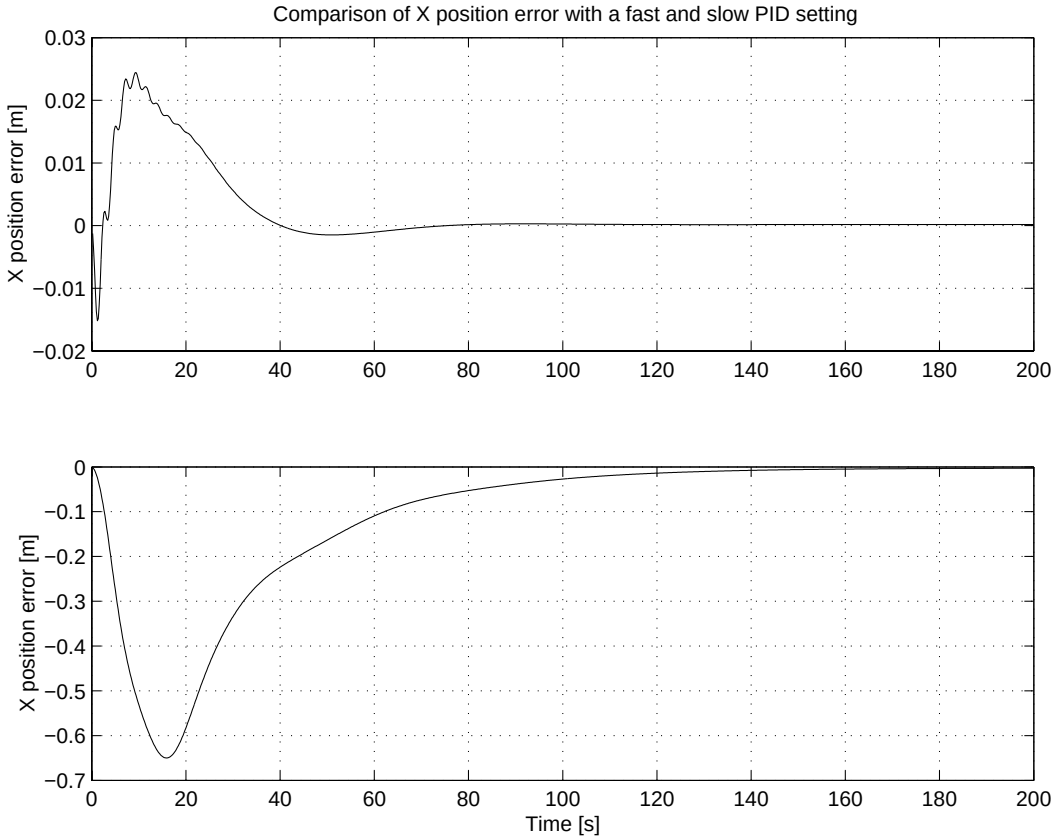


Figure 6.24: Comparison of ANGUS' X position through time when ANGUS is subject to a constant sea current $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s with a "fast" (above), and "slow" (below) set of surge PID coefficients.

since the deviation of the camera from its starting point was much greater with the slow PID controller, the tracked feature points were more likely to move out of the field of view of the camera. This was actually the case in that particular example (see figure 6.25) where the two upper right features went out of the field of view. In practice, it would have caused the servoing scheme to fail, since there were not enough features from which the homography could be computed. In this simulation however, the image plane was infinite in order to remove that constraint from the tests.

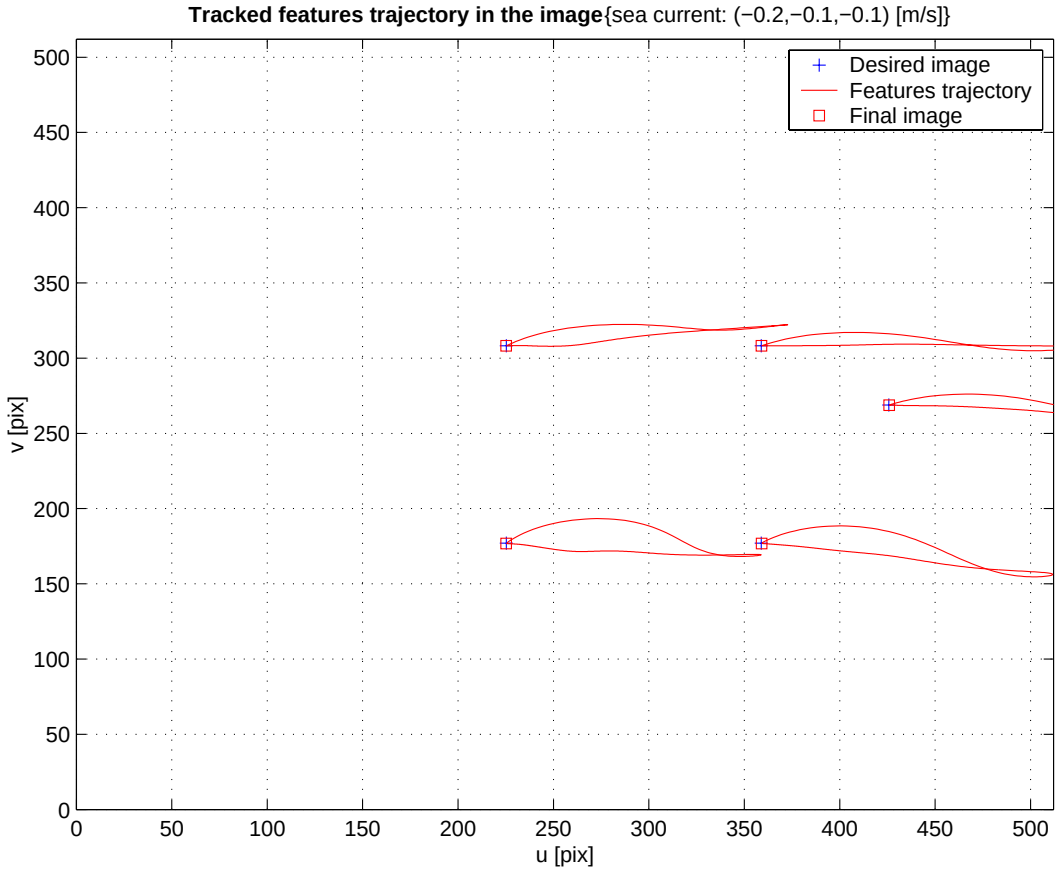


Figure 6.25: Feature points' trajectory with the "slow PID setting". The two upper right features went out of the field of view, and would have caused a servoing failure if no special strategy, such as reacquiring features if they came close to the image borders, was adopted.

The investigation of the effect that actuation noise on the thrusters' signals would have had on the visual servoing performance could have been envisioned. However, this study was not necessary because realistic actuation noise would have been outwith the robot's bandwidth, and the dynamics of ANGUS' thrusters were not modelled in the simulation that was used. Indeed, on the actual ROV, the thrusters were used exclusively in their linear characteristics by means of a limiter [3], and since the thrusters' dynamics were much faster than the ROV's, it was a good approximation to consider an affine model for the thrusters.

Conclusions

This experiment studied the effects of constant sea current disturbances on the performance of the proposed visual servoing scheme. It was shown that the station-keeping objective was met for a variety of sea current's amplitudes with a given control parameter set (the "fast PID" setting).

It was also demonstrated that the coupling between the *controlled point* and the roll and pitch angles of the camera explained the small steady-state errors in horizontal positioning.

Finally, the thrusters' ringing phenomenon was explained by the tight PID setting. This setting was necessary to obtain fast time responses to sea current disturbances. It was also required to avoid the tracked features to move out of the camera's field of view and cause a visual servoing failure. It is important to note that the tight PID setting was a design choice. Depending on the application's requirements and on the environmental conditions, the PID controller's setting can be adapted. In practice, if a slow PID controller is chosen, special strategies would need to be implemented to deal with lost features. In particular, extracting new features to keep a more or less constant number of feature points in the camera's field of view would be an option. If the lost feature point was the *controlled point*, this would need to be detected, and a new controlled point would need to be chosen. The obvious drawback of this method would be to introduce a positioning bias along the horizontal axes since the reference point would have been moved. It was however the strategy chosen for the real-time visual servoing implementation of chapter 7.

6.3.3 Influence of target orientation

In the previous simulations, it was assumed that the planar target was parallel to the focal plane of the camera. In other words, since the camera was pointing downwards, it was supposed that the target points were lying on a horizontal seabed. An issue that can be raised is whether the proposed visual control scheme would perform as well if the target points were lying on a sloping seabed. Intuitively, the performance would be expected to degrade as the slope increased until a breaking point was reached. Indeed, as the angle between the supporting plane of the tracked points and the focal plane augments, the projected points in the image move closer to one another. At the limit, i.e. when the planes cross at a right angle, the target's points project onto a single line in the image plane. Under those circumstances, the points' configuration is degenerate (see e.g. [32], pp. 74–75 for a proof). Whatever the camera's motion, there exists a family of homography transformations mapping image points from one image to the other. A direct consequence of this fact is that the more the image points approach the degenerate configuration (i.e. when the angle between the supporting plane of the focal plane becomes closer to 90°), the more ill-conditioned the homography estimation problem is. Therefore, since the pose estimation is based on the homography, it should degrade accordingly. In particular, the estimates of the ratio Z/Z^*, the normal $\mathbf{n}$ to the target plane, and $\Omega\,\mathbf{l}$ will be affected by the wrong computation of the homography.

Experimental illustration of the target's orientation influence

In the following paragraphs, the effect of the target's orientation on the station-keeping performance of the ROV simulation is examined. Consider the five 3-D points $\mathbf{M}_i$ mentioned earlier (section 6.3.1) and put them in the plane of equation $Z = 0$. Note by $\mathbf{R}$ a rotation matrix of angle θ, and rotation axis $\mathbf{u}$. To vary the angle θ between the supporting plane of the points $\mathbf{M}_i$ and the focal plane, it suffices to rotate the $\mathbf{M}_i$ with $\mathbf{R}$, and translate them along the Z-axis by 3 metres (to be consistent with the simulation setup of section 6.3.1)[12]. The transformed 3-D points $\mathbf{N}_i$ are given by:

[12]Note that all points' co-ordinates are expressed in the camera reference frame.

$$\mathbf{N}_i = \mathbf{R}\,\mathbf{M}_i + \begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix} \tag{6.30}$$

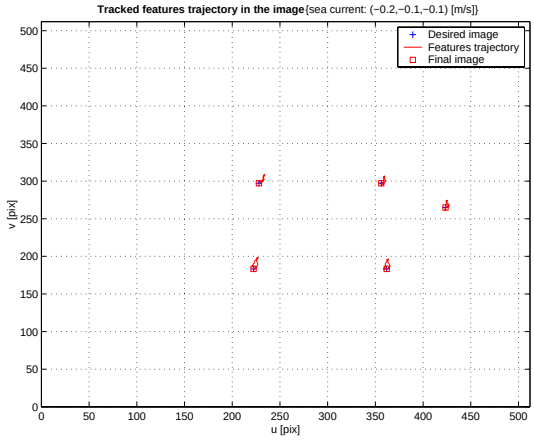
In these simulations, the 3-D points $\mathbf{M}_i$ were rotated around the camera's Y-axis by the angles: 30°, 45°, 60°, 75° and 80°. The sea current was set to $\mathbf{V}_c = [-0.2, -0.1, -0.1]^T$ m/s. Figure 6.26 illustrates the effect, in the image plane, of the variation of the target's orientation on the projected 3-D points. As the angle θ increased, the pentagon formed by the five projected points tended to transform into a line, a degenerate configuration.

Figure 6.27 collects the positioning errors along the X, Y and Z axes. Several remarks can be made. The settling time augmented as the target inclination increased. This was very clear on the positioning error in Z, which was the most affected by the target's inclination. For instance, when $\theta = 75^\circ$, the positioning error in Z exhibited a small but noticeable steady-state error (roughly 2 mm). This angle was so chosen because it was the limit angle of convergence. Indeed, for angle's values greater than 75°, the position in X, Y, Z all exhibited steady-state errors. An illustration of the loss of convergence to within a few millimetres, can be observed in figure 6.27 (e), when $\theta = 80^\circ$. This figure shows steady-states errors of 6.5 cm in X, 4.5 cm in Y, and up to 40 cm in Z! For angles greater than 80°, the system became unstable.

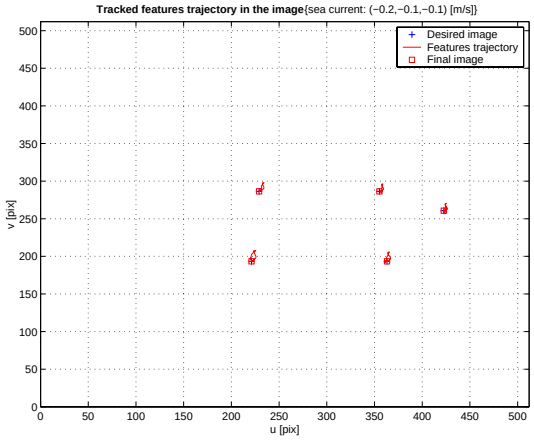
Similar remarks can be made on the orientation errors shown in figure 6.28. The time response of the robot increased as the target orientation augmented. Noticeable steady-state errors appeared when the target's orientation reached 75° and the visual control stopped converging, to within a few millimetres, at angles greater than 75°. Above 80° the system became unstable.

The heave d.o.f. was the most affected by the target's orientation. Indeed, the ratio between the desired value along the Z axis, Z^*, and the actual value Z is given by the determinant of the estimated homography matrix. The estimation of this value was an ill-posed problem when the target's orientation approaches 90°.

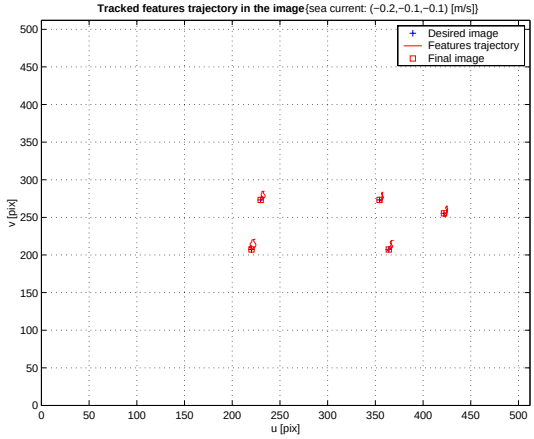
Despite the undesirable behaviour of the visual control scheme when the target's orientation is important (greater than 75°), it is necessary to point out that such



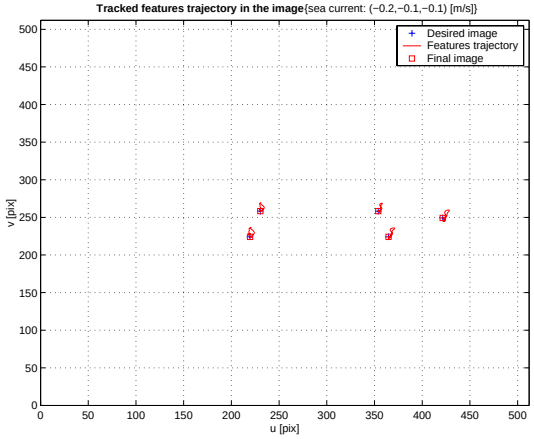
(a) Target orientation: 30 °



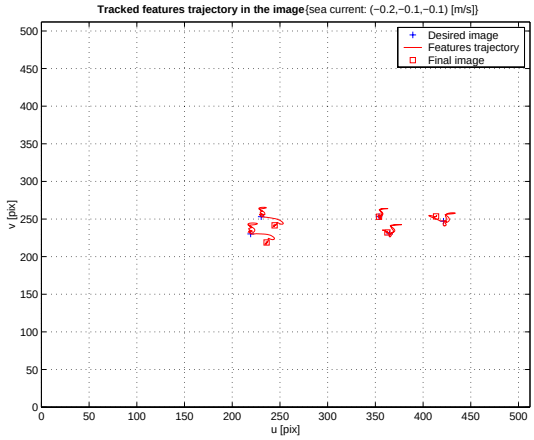
(b) Target orientation: 45 °



(c) Target orientation: 60 °

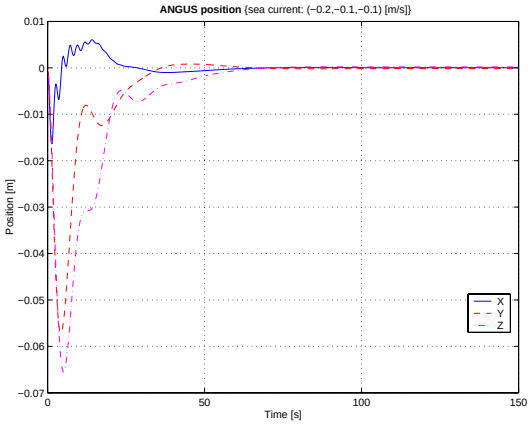


(d) Target orientation: 75 °

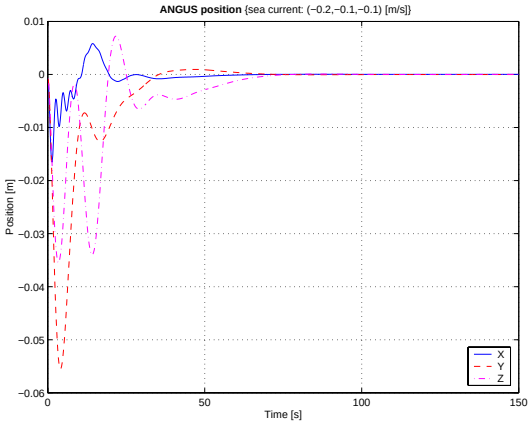


(e) Target orientation: 80 °

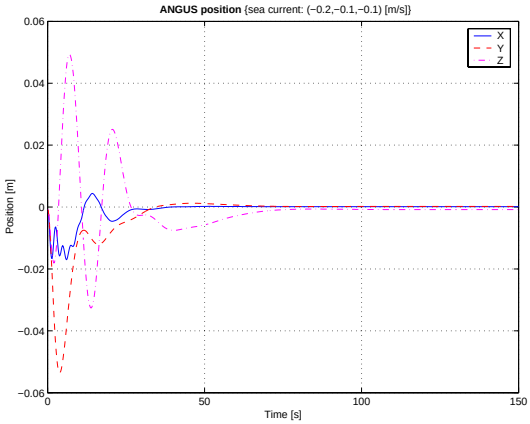
Figure 6.26: Snapshots of the pixel trajectory with a varying target orientation.



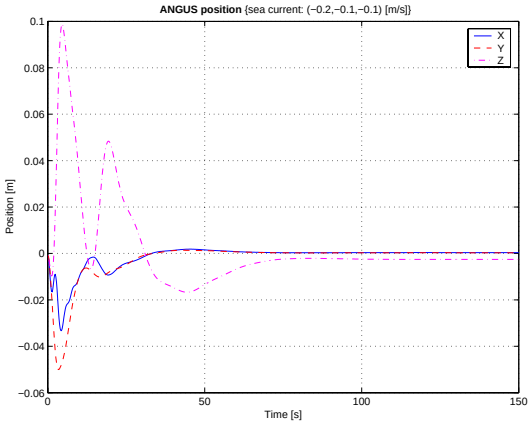
(a) Target orientation: 30 °



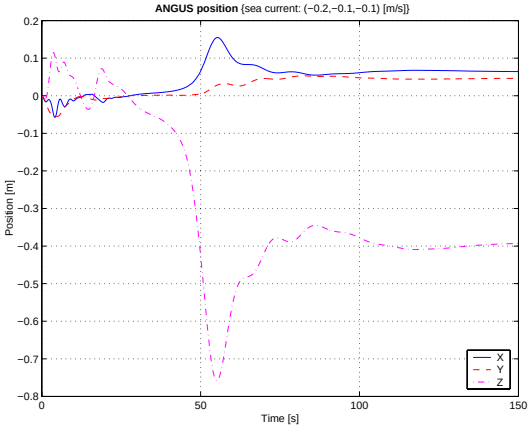
(b) Target orientation: 45 °



(c) Target orientation: 60 °



(d) Target orientation: 75 °



(e) Target orientation: 80 °

Figure 6.27: Influence of the target orientation on robot's positioning errors. From (a) to (d), the time response of the robot increases. Figure (d) exhibits more noticeable steady-state errors. Finally, plot (e) shows the loss of convergence of the visual control.

slopes are seldom met on natural seabeds. So, it is a limitation of the algorithm, but without real consequences on its exploitation at sea.

Conclusions

The proposed visual scheme proved to be robust with a large range of target plane's slant. Problems do occur for slopes greater than 75°. However, such slopes are unlikely to be encountered underwater. A typical slope value of an underwater mountain is about 30° (see e.g. Mount Therese, off South East Ireland).

6.3.4 Influence of noise from the feature tracker

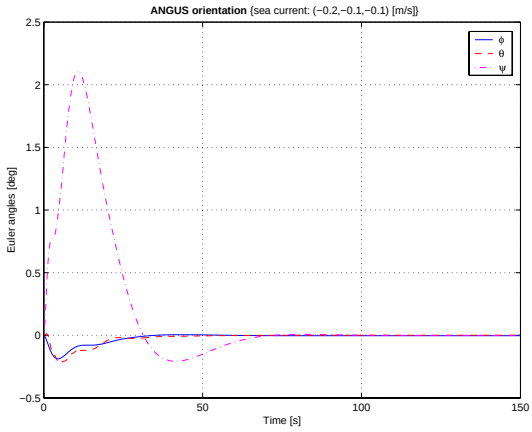
This set of simulations aimed at assessing the robustness of the algorithm with respect to noise from the feature tracker. White Gaussian noise was added to corrupt the 3-D position of the five target points. The procedure was the following. At each iteration k of the control loop, each 3-D point $\mathbf{M}_i$, $i \in [1, 5]$ was corrupted with noise of zero mean and standard deviation in metre, σ_m. Let the realisation of this noise be $n(k)$. Another random variable following a zero mean Gaussian distribution of standard deviation 2π: $\theta(k)$ was also defined. The noisy 3-D points (denoted by a bar) were then chosen as:

$$\forall\, i \in [1,5] : \begin{cases} \bar{X}_i(k) & = X_i(k) + n(k)\, \cos\left(\theta(k)\right) \\ \bar{Y}_i(k) & = Y_i(k) + n(k)\, \sin\left(\theta(k)\right) \\ \bar{Z}_i(k) & = Z_i(k) + n(k). \end{cases} \tag{6.31}$$

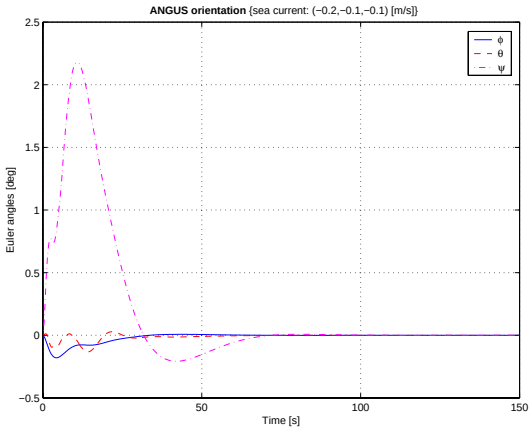
The noise in location of the target points was converted to its corresponding noise in the image using the simple formula:

$$\sigma_{pix} = f\frac{\sigma_m}{d^*} = \frac{800}{3}\sigma_m$$

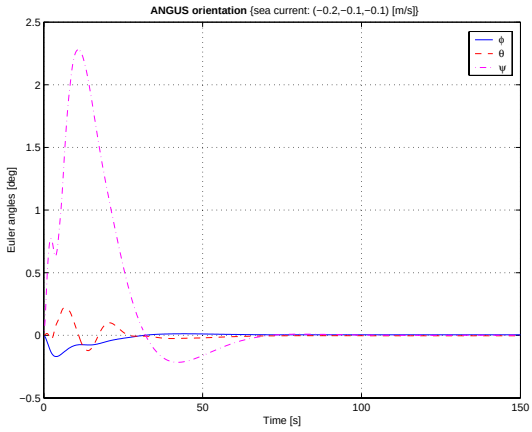
where the focal length of the camera was assumed to be $f = 800\,\text{pix/m}$ (see table 5.1).



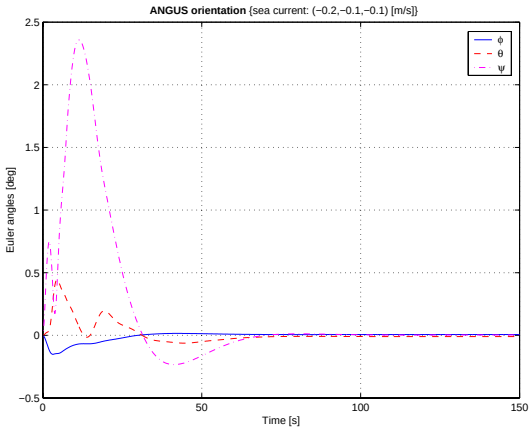
(a) Target orientation: 30 °



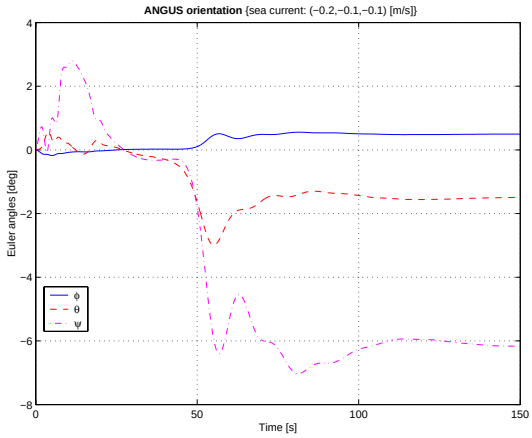
(b) Target orientation: 45 °



(c) Target orientation: 60 °



(d) Target orientation: 75 °



(e) Target orientation: 80 °

Figure 6.28: Influence of the target orientation on robot's orientation errors. From (a) to (d), the time response of the robot increases. Figure (d) exhibits more noticeable steady-state errors. Finally, plot (e) shows the loss of convergence of the visual control.

The simulations were run for the case of a moderate strength sea current of velocity $\mathbf{V}_c = [-0.6, -0.2, -0.2]^T$ m/s. For each noise level σ_{pix}, an experiment would run for 5,000 iterations. The mean deviation of the robot's position with respect to its desired hover position was computed from iteration 500 until the end. This ensured that the robot would have stabilised in the absence of noise. Each measurement was therefore computed over a real-world time of 15 minutes[13]. The noise level was gradually increased from $\sigma_{pix} = 0$ to a maximum of $\sigma_{pix} = 8$ pixels. The robot was able to hover around the desired position within a 50 cm radius with a noise level as high as 5 pixels (see figures 6.29, 6.30, and 6.31 for the errors on the X, Y and Z axes respectively). Even better results were obtained with respect to the heading of the UUV. Indeed, the heading remained within $\pm 1^\circ$ for the same noise level. The heading error is shown on figure 6.32.

The results lead to the conclusion that the estimation of the rotation from the homography is less affected by noise than the estimation in translation. They also show that the robot did not drift, therefore satisfying, to within a given accuracy, the requirements of a station-keeping system.

[13]With a sampling period of $Ts = 0.2$ s, 4,500 iterations represented 900 seconds, or 15 minutes.

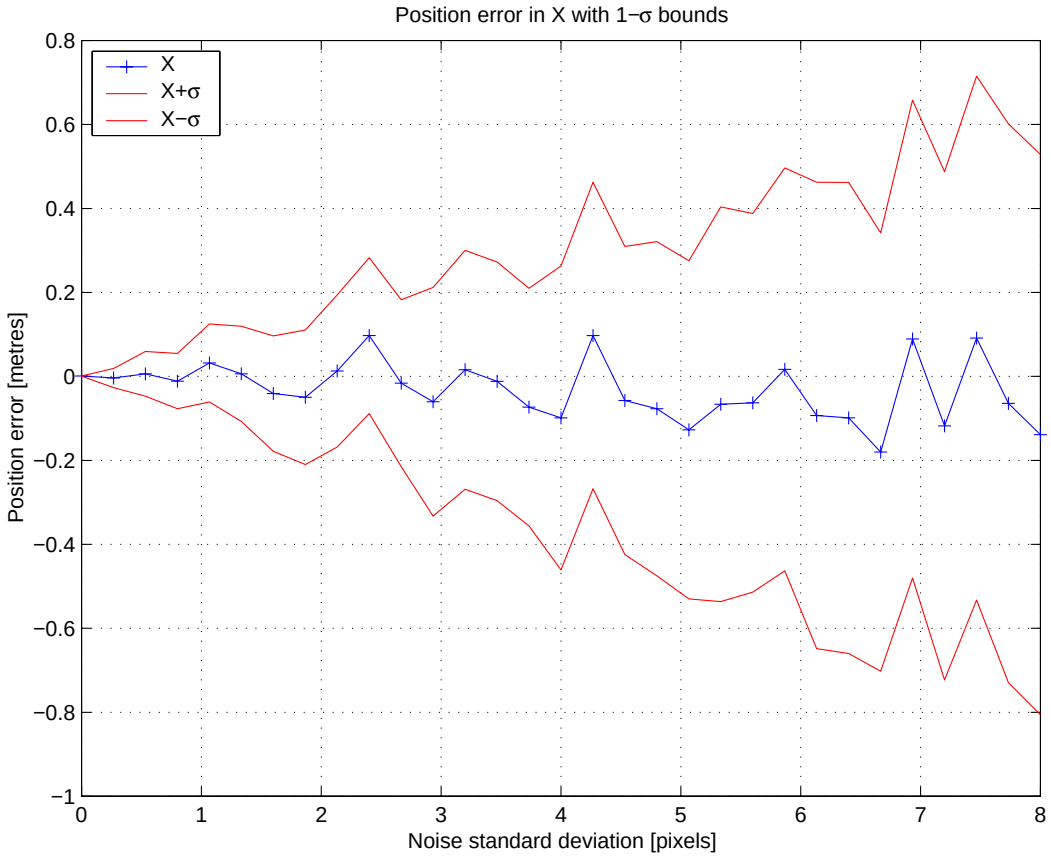


Figure 6.29: Influence of tracking noise: mean hovering error for the X-axis, and 1-σ bound.

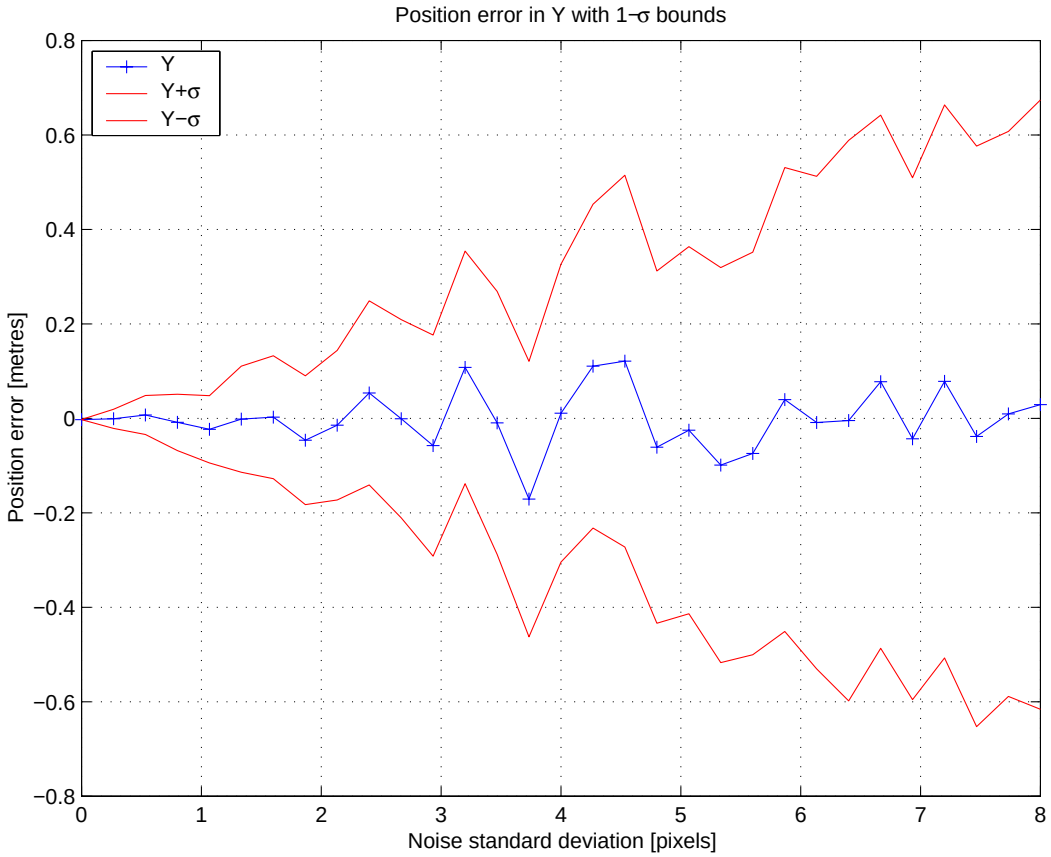


Figure 6.30: Influence of tracking noise: mean hovering error for the Y-axis, and 1-σ bound.

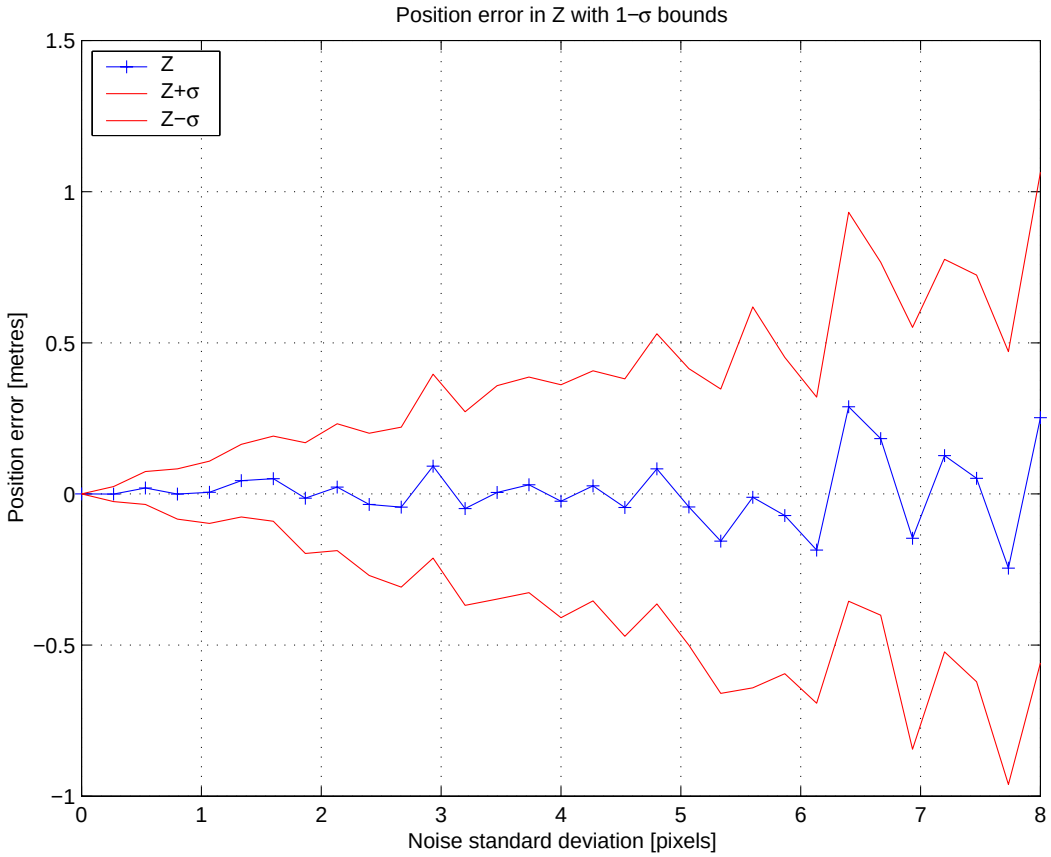


Figure 6.31: Influence of tracking noise: mean hovering error for the Z-axis, and 1-σ bound.

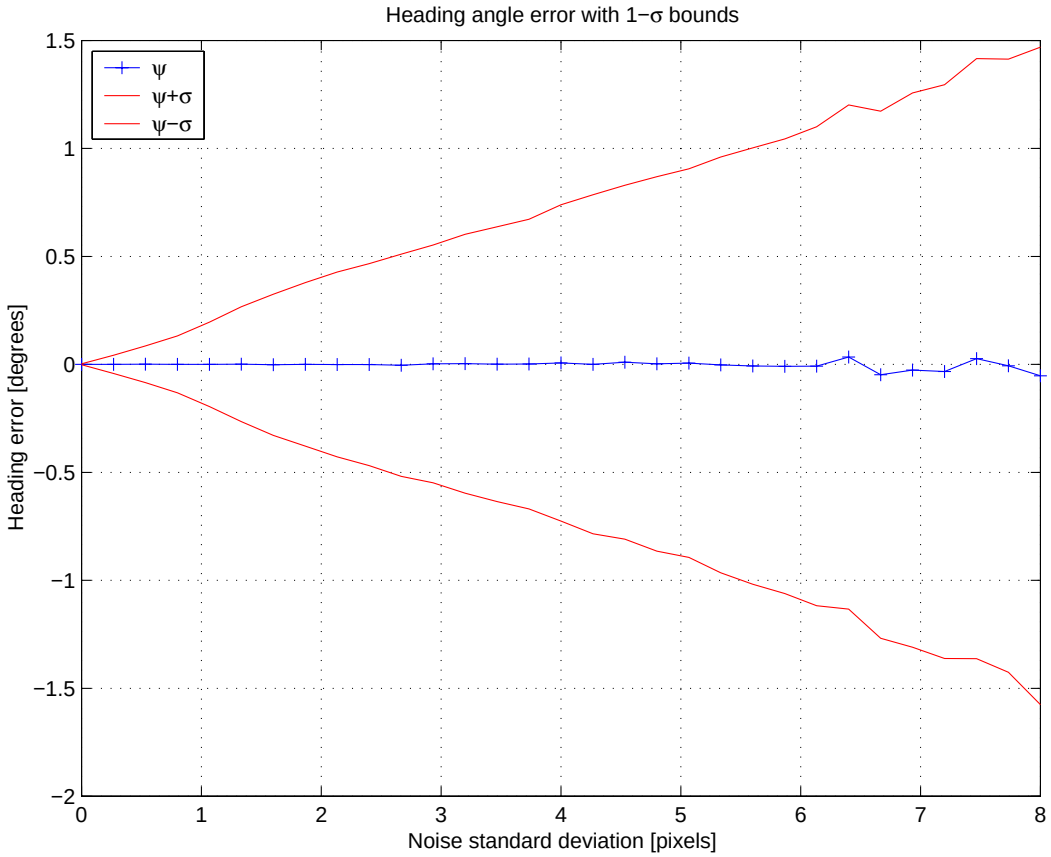


Figure 6.32: Influence of tracking noise: mean hovering error for heading, and 1-σ bound.

6.4 Conclusions

This chapter tackled visual station-keeping of a 6 degrees-of-freedom underwater vehicle.

Originally based on the 2 1/2 D technique, the proposed visual servoing algorithm took into account the underactuation of a typical underwater vehicle. The method employed to consider the uncontrollable degrees-of-freedom of the UUV can be easily adapted to other types of robots.

The simulation results demonstrated that:

- The proposed approach was robust to noise from the feature tracker.
- The control was robust to various sea disturbances.
- The proposed scheme also worked on steep seabed slopes.

For the homography estimation, the linear algorithm described in chapter 4 was used. As with any linear estimation algorithm, it is sensitive to noise. Although it can cope quite well with white noise, outliers in the feature matching would produce biased results. In this case, a statistical estimator such as RANSAC or LMedS [54] could be used to eliminate outliers. Alternatively, removing outliers in the tracking part of the visual servoing process would probably lead to similar results.

The main concern was to prove the feasibility of visual servoing underwater vehicles. For that reason, a straightforward PID control was applied, rather than a more advanced controller. A better time response would be expected if a more appropriate control algorithm were to be used.

The use of the ANGUS 003 dynamic model gave some good clues on how an underwater vehicle would behave. However, although the dynamic model was complex: nonlinearities, cross-coupling, and its hydrodynamic parameters were experimentally evaluated, it was still an approximation of the reality. Consequently, further work would involve the validation of the approach on a real underwater vehicle. The next chapter will present a first step towards an experimental validation of the proposed approach.

Chapter 7

A 2-D visual servoing technique for underwater vehicle station keeping

The main objective of this chapter[1] was to demonstrate the feasibility of visual servoing for underwater vehicles in practice. More specifically, the aim was to demonstrate that the feature tracker evaluated in chapter 5 could be used for the visual station keeping of a **typical** underwater vehicle. For that purpose, the behaviour of a typical UUV based on a simulation of the ROV ANGUS on a Cartesian robot was emulated. This simulation was within the limits exposed in chapter 3. However, the simulation had only two controllable horizontal d.o.f., therefore, only a reduced order model of ANGUS could be used to validate the proposed approach. The visual servoing task of chapter 6 was therefore re-designed to better suit the problem at hand.

In addition, it was possible to evaluate the positioning accuracy of visual servoing in a water tank with the same experimental setup of chapter 5. Since, as was discussed in chapter 1, positioning sensors for underwater vehicles are not accurate enough for the station-keeping process evaluation, the use of the Cartesian robot permitted this quantitative assessment.

This chapter is organised as follows. First, a reduced order dynamic model of ANGUS is presented. This model, including only the surge and sway degrees-of-freedom, was derived from the higher order model presented in chapter 2. Then, a 2-D visual servoing task is proposed to solve the dynamic positioning problem. The chapter closes by discussing the results of real-time experiments in the water tank.

[1] The work presented in this chapter is a revised version of the paper [47].

7.1 A reduced order dynamic model of ANGUS

To experimentally demonstrate the station-keeping capabilities on the Cartesian robot, its two d.o.f., which emulated the surge and sway axes of ANGUS, must be constrained.

Let the body-fixed vehicle's velocity vector be defined as $\boldsymbol{\nu} = [u, v]^T$, and the Earth-fixed velocity vector as $\dot{\boldsymbol{\eta}} = [\dot{x}, \dot{y}]^T$. It was assumed that the vehicle's heading angle ψ remained constant and close to zero (for example using an independent control loop) [46]. In addition, since ANGUS was designed to be stable in roll (θ) and pitch (ϕ), these angles also remained very small while hovering. As a result, the 2×2 Jacobian matrix relating the body-fixed velocity vector $\boldsymbol{\nu}$ to the Earth-fixed velocity vector $\dot{\boldsymbol{\eta}}$ could be approximated by the 2×2 identity matrix (first-order Taylor expansion). In the following, the distinction between Earth-fixed position and velocity, and body-fixed ones will no longer be made. The motion equations for the ROV can thus be written as:

$$\begin{aligned} M_{11}\dot{u} &= B_{11}(u - u_c)(|u - u_c| + D_u) + a_1\beta + a_2|\beta| \\ M_{22}\dot{v} &= B_{22}(v - v_c)(|v - v_c| + D_v) + a_3\gamma \end{aligned}$$

where M_{ii} $(i = 1, 2)$ are the mass matrix coefficients, the B_{ii} $(i = 1, 2)$, D_u and D_v are the hydrodynamic drag coefficients, u_c and v_c are the velocity of the sea current expressed in the body-fixed frame. The normalised control input of the two back thrusters is $\beta \in [-1, 1]$, while $\gamma \in [-1, 1]$ is the normalised control input of the sway thrusters, and a_i $(i = 1, 2, 3)$ are the thrusters' efficiency coefficients. Note that the back thrusters are less efficient when operated in reverse. The numerical values of the parameters are gathered in table 7.1.

To assess the dynamic performances of ANGUS, a "ground truth" PID controller was designed assuming perfect position and velocity measurements at the same sampling rate as the visual servoing experiments, i.e. $T_s = 200\,\text{ms}$. Also included was the time delay of one sampling period caused by the visual processing. The PID control law was:

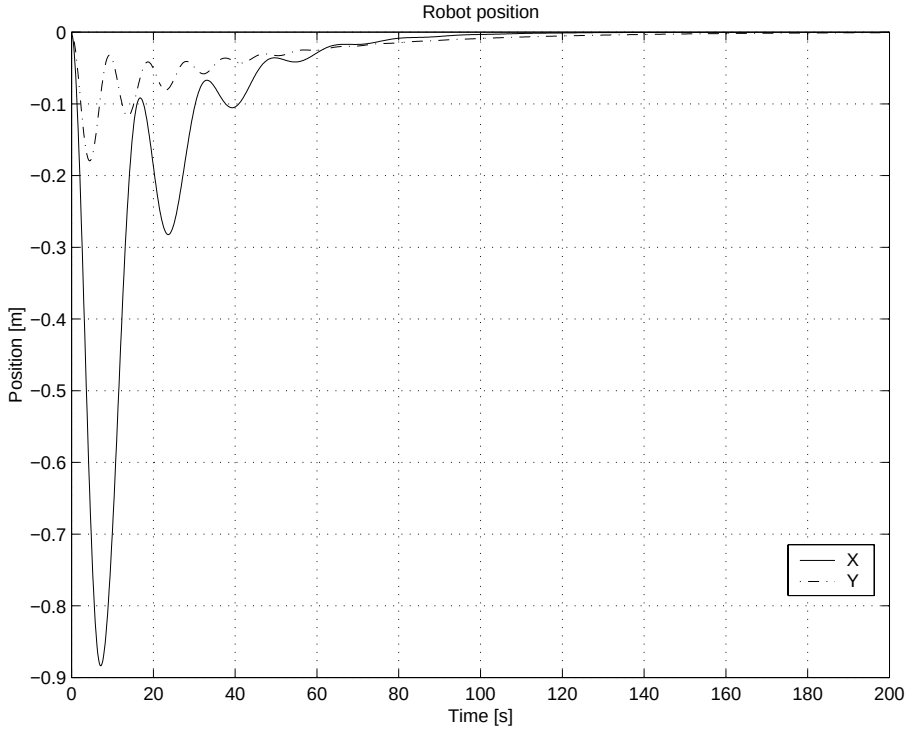


Figure 7.1: Positioning error of the 2 d.o.f. model of ANGUS subject to a sea current disturbance $(u_c, v_c) = (-0.2, -0.5)\,\mathrm{m/s}$ with a PID controller.

$$\begin{bmatrix} \beta \\ \gamma \end{bmatrix} = \mathbf{K}_p \begin{bmatrix} x \\ y \end{bmatrix} + \mathbf{K}_d \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} + \mathbf{K}_i \begin{bmatrix} \int_0^t x dt \\ \int_0^t y dt \end{bmatrix} \tag{7.1}$$

where $\mathbf{K}_p = \mathrm{diag}(1.0, 4.0)$, $\mathbf{K}_d = \mathrm{diag}(0.4, 0.4)$, and $\mathbf{K}_i = \mathrm{diag}(0.01, 0.02)$.

Since surge and sway were decoupled, the parameters of each axis were tuned independently. The design goal was to obtain a time response similar to a second order critically damped system. In figure 7.1, note the slow dynamic response of the ROV subject to a sea current velocity step input disturbance of $(u_c, v_c) = (-0.2, -0.5)\,\mathrm{m/s}$. The settling times, corresponding to a positioning error of less than one centimeter, of the surge and sway axis are: $t_{surge} = 80\,\mathrm{s}$ and $t_{sway} = 95\,\mathrm{s}$. Figure 7.2 shows the corresponding thrusters' values. These numbers will serve as a basis of comparison for the visual servoing experiments.

In the following section, an image-based visual servoing approach to solve the station keeping problem is presented. It is a simple 2-D visual servoing scheme [20] using the centre of gravity of the set of extracted features in the images as the visual feature. Since the camera was rigidly mounted onto the robot, the frame transformation

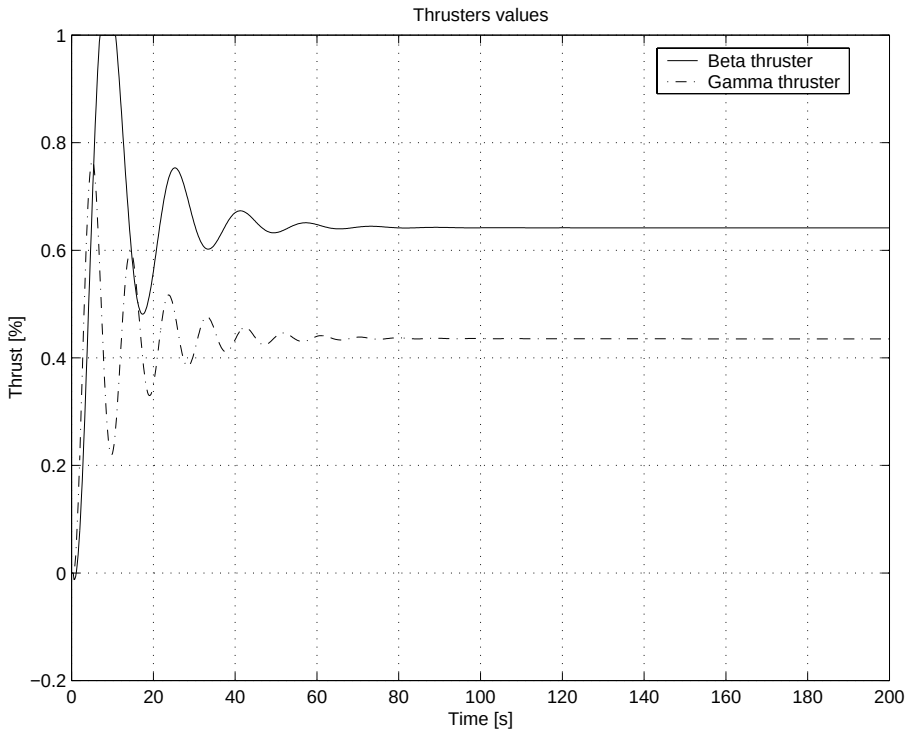


Figure 7.2: Thrusters' values of the 2 d.o.f. model of ANGUS subject to a sea current disturbance $(u_c, v_c) = (-0.2, -0.5)$ m/s with a PID controller.

M_{11}	M_{22}	B_{11}	B_{22}	D_u	D_v	a_1	a_2	a_3
1800	2200	-350	-680	0.6	0.6	500	200	250

Table 7.1: Dynamic model parameters values

between the robot frame and the camera frame was therefore constant. To simplify the expression of the following equations, the camera reference frame and the robot frame were assumed to coincide. In other words, controlling the robot was equivalent to controlling the camera whose motion dynamics were close to those of a typical underwater vehicle.[2]

7.2 The 2-D visual servoing task

In section 6.1.2, it was seen that a visual servoing positioning task could be expressed as the regulation to zero of a task function $\mathbf{e}(\mathbf{s}, t)$, where s is a visual feature, and t is the time [72]. If $\mathbf{T}_c$ is the camera velocity screw (6-vector), and $\mathbf{L}(\mathbf{s}, t)$ is an

[2]Within the limits exposed in chapter 3.

appropriate matrix, called the *interaction* matrix, then the camera velocity screw is related to the task function $\mathbf{e}$ by:

$$\dot{\mathbf{e}} = \mathbf{L}\,\mathbf{T}_c \tag{7.2}$$

In the case of 2-D visual servoing, if $\mathbf{s}_i = [x_i, y_i]^T$ is a feature point expressed in the image (Z_i being the depth of the corresponding 3-D point projected on the image point $\mathbf{s}_i$), $\mathbf{L}$ is made up by stacking the following lines (two lines for each feature point):

$$\begin{bmatrix} -1/Z_i & 0 & x_i/Z_i & x_i y_i & -(1+x_i^2) & y_i \\ 0 & -1/Z_i & y_i/Z_i & 1+y_i^2 & -x_i y_i & -x_i \end{bmatrix} \tag{7.3}$$

An exponential decrease of the task function is obtained by imposing $\dot{\mathbf{e}} = -\lambda\,\mathbf{e}$ (where λ is a positive scalar) so that the corresponding control law would be

$$\mathbf{T}_c = -\lambda \mathbf{L}^{+}\mathbf{e}, \tag{7.4}$$

if the interaction matrix $\mathbf{L}$ is not singular. For instance, if four 3-D points forming a square are considered, the interaction matrix made from the projections of these four points is non-singular (rank 6). The interested reader is referred to F. Chaumette's PhD thesis [5] for an expression of $\mathbf{L}$ in that case.

As the goal is to constrain the two translational d.o.f. of the camera parallel to the image plane, the information provided by a single feature point is sufficient. In any case, for robustness purposes, the centre of gravity $[x_g, y_g]^T$ of the set of extracted features was chosen. In this case, the interaction matrix had the simple form:

$$\mathbf{L} = \begin{bmatrix} -1/Z_g & 0 \\ 0 & -1/Z_g \end{bmatrix} \tag{7.5}$$

If $\mathbf{s}_g^* = [x_g^*, y_g^*]^T$ is the position of the centre of gravity of the features set in the initial (*desired*) position, and $\mathbf{s}_g = [x_g, y_g]^T$ in the *current* one, $\mathbf{e}$ can be defined as $\mathbf{e} = \mathbf{s}_g - \mathbf{s}_g^*$. Therefore, when $\mathbf{e}$ is zero, the camera is in its desired position. A proportional control law which regulates the camera at its desired position reads as

eq. (7.4). To be able to reject constant sea current disturbances, since the closed loop system is of type 0, an integration term in the control law was needed. It was decided to implement a vectorial PID controller to obtain a better stability than a simple PI controller. The proposed control law was then:

$$\begin{bmatrix} \beta \\ \gamma \end{bmatrix} = \begin{bmatrix} -Z_g & 0 \\ 0 & -Z_g \end{bmatrix} (\mathbf{K}_p \, \mathbf{e} + \mathbf{K}_d \, \dot{\mathbf{e}} + \mathbf{K}_i \int_0^t \mathbf{e} \, dt) \tag{7.6}$$

where $\mathbf{K}_p$, $\mathbf{K}_d$, and $\mathbf{K}_i$ are 2×2 diagonal matrices with the proportional, derivative and integration control gains of the PID, and Z_g is the depth of the centre of gravity in the current image, estimated from:

$$Z_g \approx Z_g^* \, \det(\mathbf{H}). \tag{7.7}$$

The experimental results obtained with this proposed visual servoing algorithm are now described.

7.3 Experimental results

7.3.1 Experimental protocol

Each station keeping experiments followed the same procedure. The robot was initially immobile, and the desired set of features was extracted from the first image taken from the initial position. A constant sea current disturbance (step input) of velocity (u_c, v_c) was applied. As the robot started drifting away from its desired position, the set of features was tracked in the current image. The depth Z_g was then estimated from the homography between the initial set of features and the current set of features (eq. 7.7 as described in section 4.3.3. Similarly, the task function $\mathbf{e}$ was calculated using the centre of gravity of the current feature set. If a feature was lost during tracking, its corresponding feature in the initial set was removed. As long as at least four features remained tracked, the servoing could continue (four points, of which no three are collinear, are needed for the estimation of an homography).

For the whole set of experiments, the control gains of the PID remained constant with the following values: $\mathbf{K}_p = \text{diag}(1.0, 2.0)$, $\mathbf{K}_d = \text{diag}(2.0, 2.0)$, and $\mathbf{K}_i = \text{diag}(0.01, 0.01)$. The estimated distance to the target Z_g^* was roughly measured and set to 1.1 m.

Experiments with different sea current amplitudes are reported in this chapter. For each experiment, the position and velocity of the Cartesian robot, the values of the task function $\mathbf{e}$, the error in features' co-ordinates, the thrusters values applied to the ANGUS model, and the features' trajectory in the image were plotted. In addition, a set of grabbed and processed images associated to each experiment were included, such as in figure 7.9 for instance.

7.3.2 Nominal experiments

In this section, the results of four **successful** station-keeping experiments are presented and analysed.

In the first experiment, the sea current disturbance was set to $\mathbf{V}_c = [u_c, v_c]^T = [-0.1, -0.1]^T$ m/s. The positioning errors of the Cartesian robot along both axes are shown in figure 7.3. As expected, after an initial deviation induced by the sea current's action on ANGUS' simulation model, the servoed system came back within a close vicinity of its initial position. Indeed, in this experiment, the steady-state errors were of 5 mm for X and 2.5 mm for Y. From a theoretical point of view, it was not a steady-state error since there was a remaining minute drift on the positions' signals. However, the residual speed measured was stable and null within the robot's positioning accuracy (see figure 7.4). Indeed, for both axis, the mean speed value for the last 100 samples was of the order of one tenth of a millimeter per second with a standard deviation smaller than 0.4 mm/s. Since the Cartesian robot's positioning accuracy was of 0.8 mm, the measured speed values were too minute to be measured, and it was more likely that noise on the encoders' was measured instead. Therefore the visual servoing scheme was stable and exhibited steady-state positioning errors from a practical point of view.

Now, since the PID controller included an integral action, the closed-loop system

would have been expected to reject perfectly constant disturbances. The positioning steady-state errors contradicted this claim. It is, however, easily explained by figure 7.8, where the feature points' tracks in the image are collected. At the end of the experiment, some tracked features' locations were far from their desired location (see for instance the fourth central feature, counting from the bottom of the image). This phenomenon is also noticeable in figure 7.6 where steady-state errors of up to 20 pixels on the horizontal axis can be seen. The steady-state errors in the feature points' locations propagated in the task function (see figure 7.5) since it used the centre of gravity of the feature points and finally these errors propagated to the robot's positions.

Although some of the features were badly tracked (up to 20 pixels error on one of them!), the robot's positioning errors remained within a few millimetres. This was a direct consequence of designing the task function using the centre of gravity of the tracked features. It amounted to average the measurements of all the features' displacements, therefore reducing the impact of outliers, and thus the bias on the estimated position of the centre of gravity.

The errors in tracking the feature points' were the consequence of the drifts of the integration. Drift of the feature tracking algorithm has been experimentally evaluated in chapter 5.

For the sake of clarity, selected snapshots of the grabbed images on which the feature tracking was performed are gathered in figure 7.9. The visual target was, as claimed, unstructured (or natural). In this particular experiment it was mainly gravel lying on the concrete bottom of the water tank. Incidentally, note that the two upper leftmost features were lost during the experiment since they went out of the field of view. This did not affect the station-keeping's objective.

The normalised demanded thruster signals were plotted in figure 7.7. Noise in the estimation of the task function was propagated onto the thruster signals. It did not, however, affect adversely the outcome of the experiment. Besides, the robotic system comprised of the Cartesian robot combined with ANGUS' simulation acted as a low pass filter, hence lessening the noise effect.

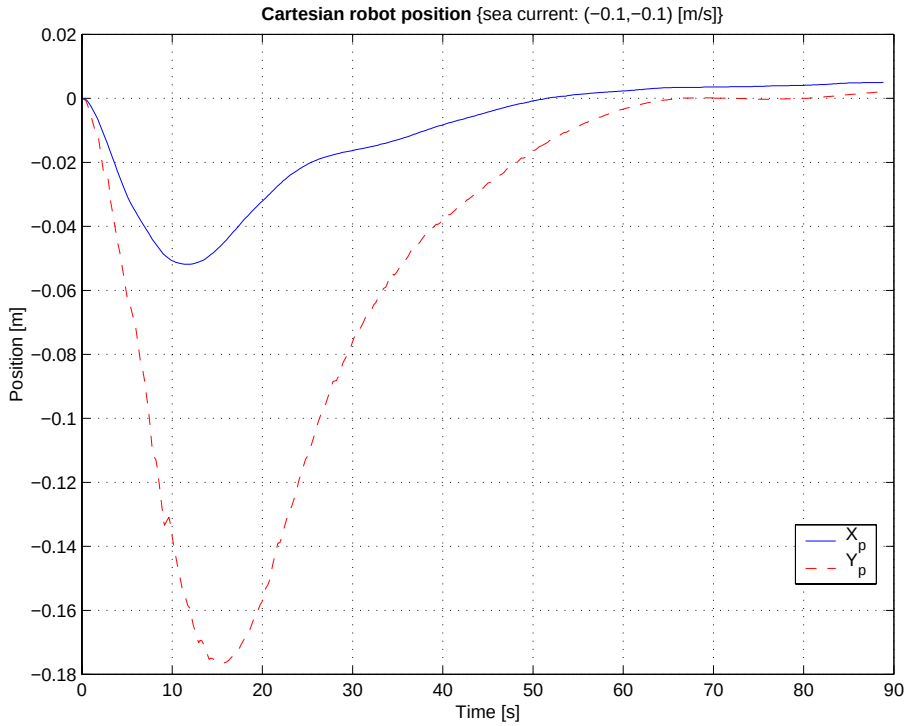


Figure 7.3: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: Cartesian robot's position.

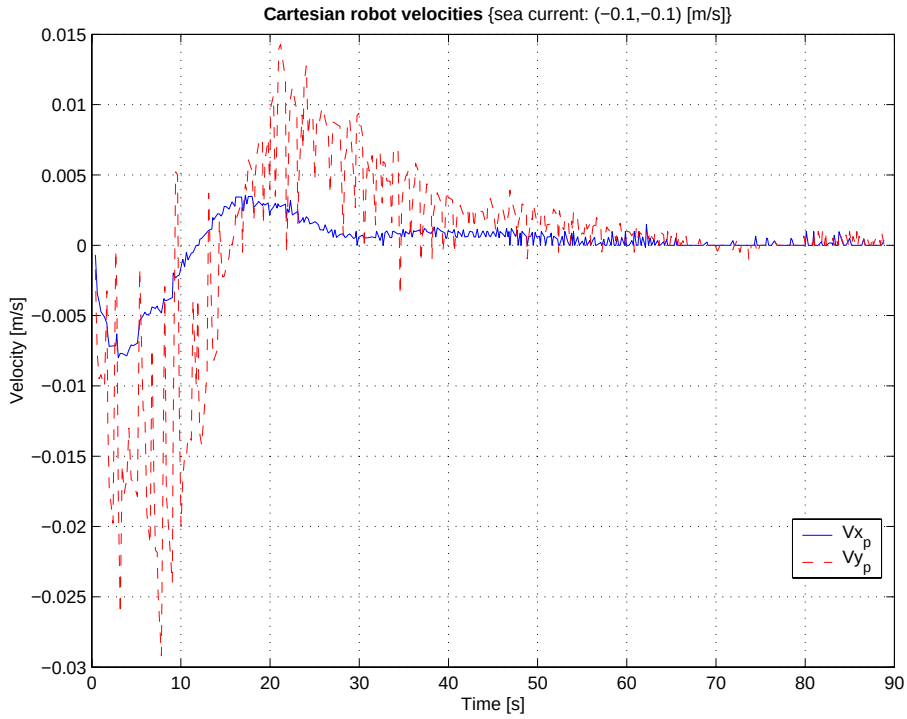


Figure 7.4: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: Cartesian robot's velocities.

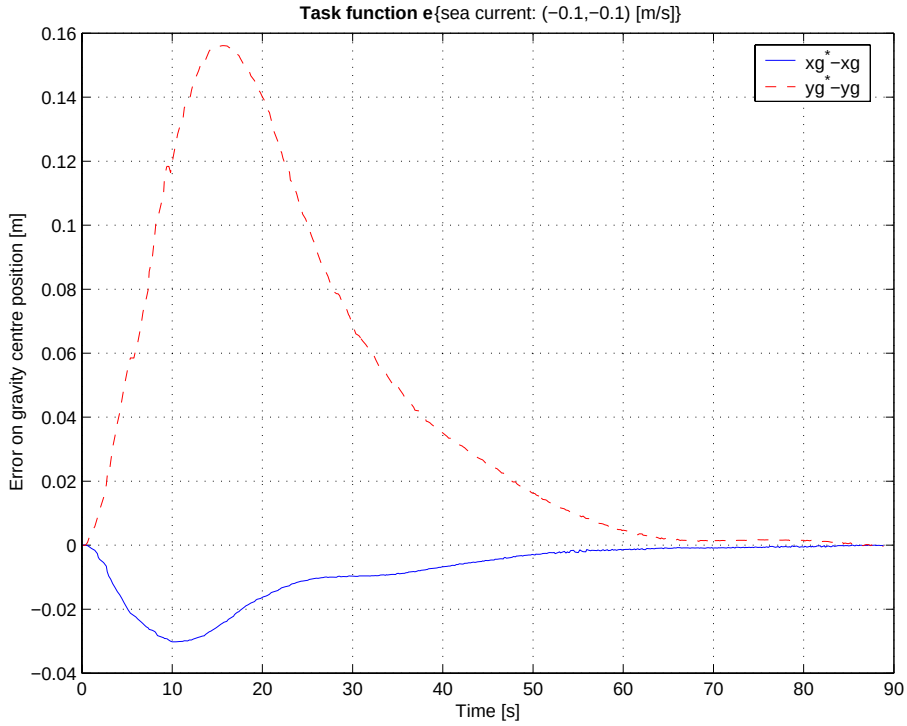


Figure 7.5: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: task function.

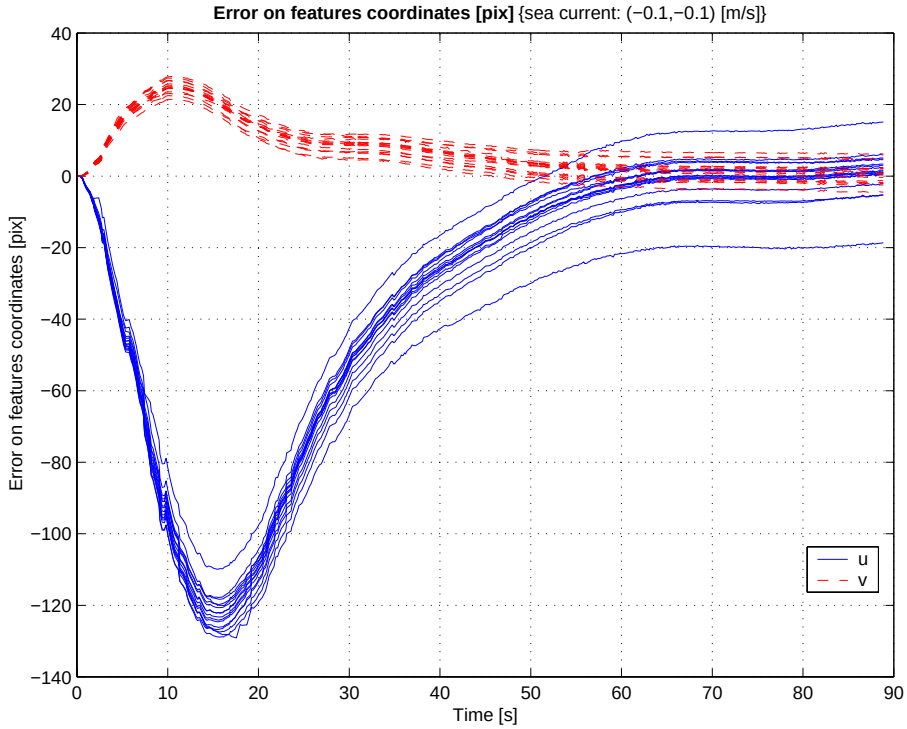


Figure 7.6: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: error on features' co-ordinates.

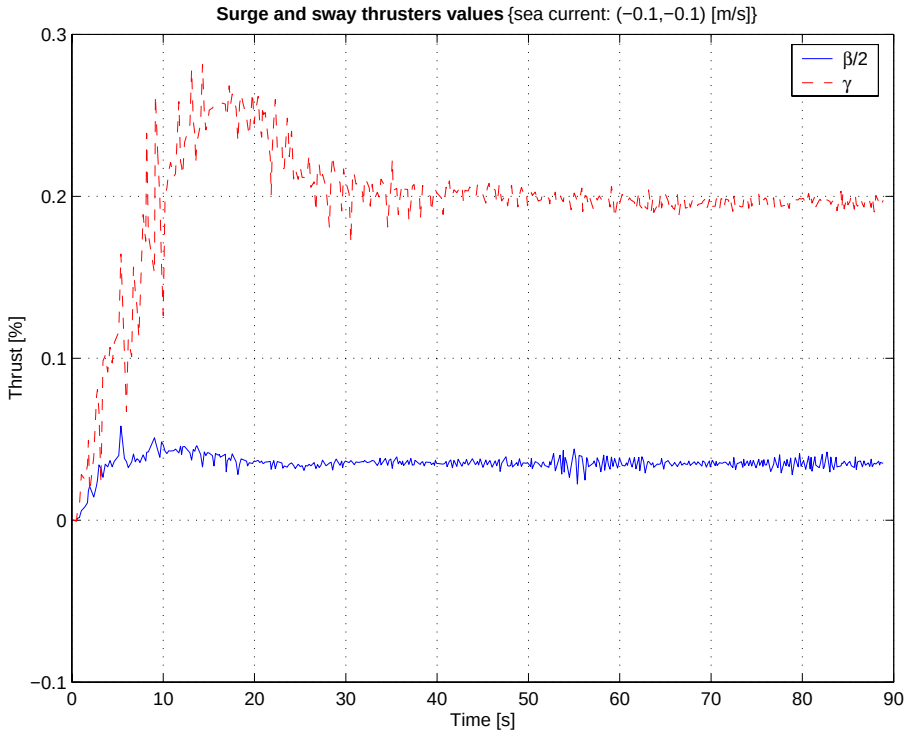


Figure 7.7: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: surge and sway thrusters.

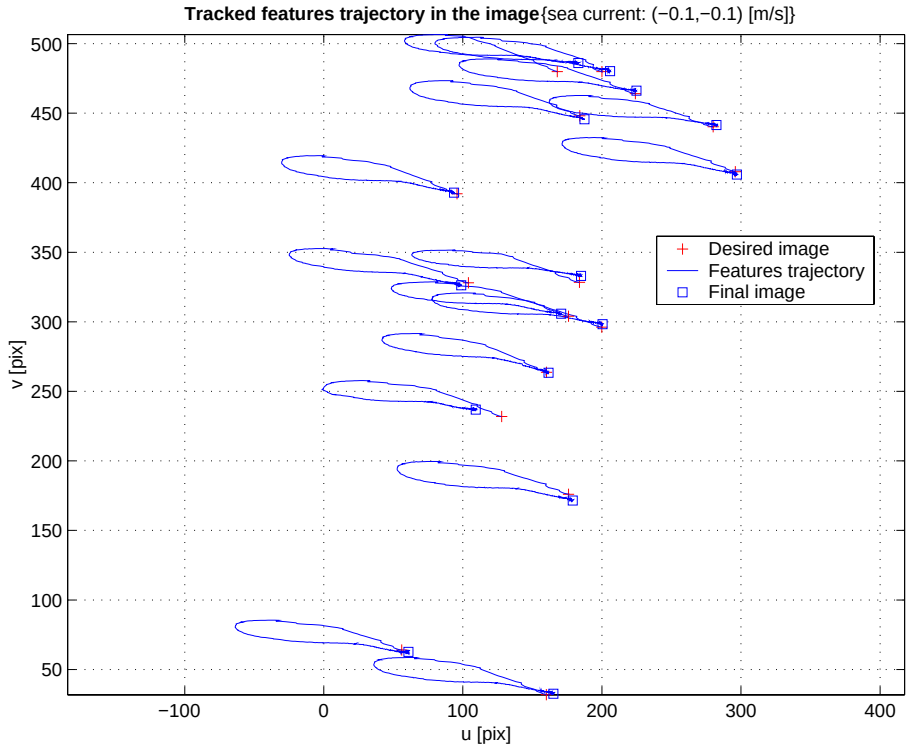
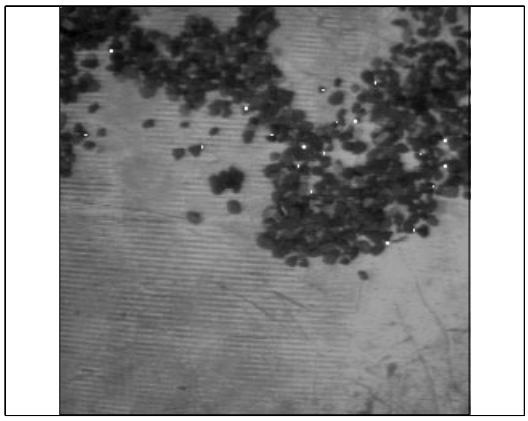
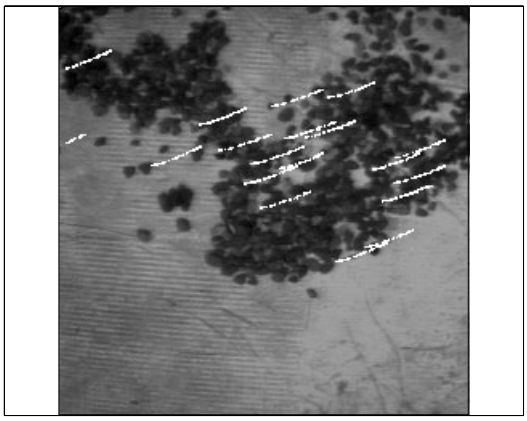


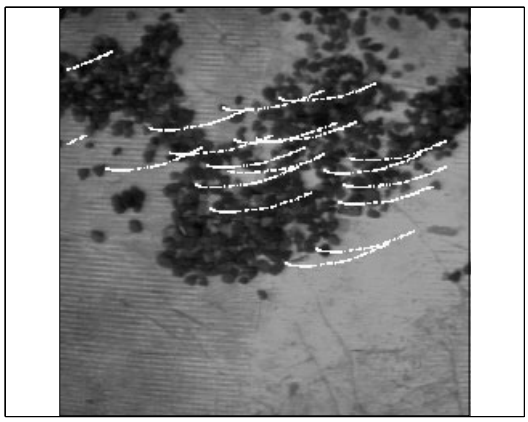
Figure 7.8: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: trajectory of the features in the image



(a) Image 0



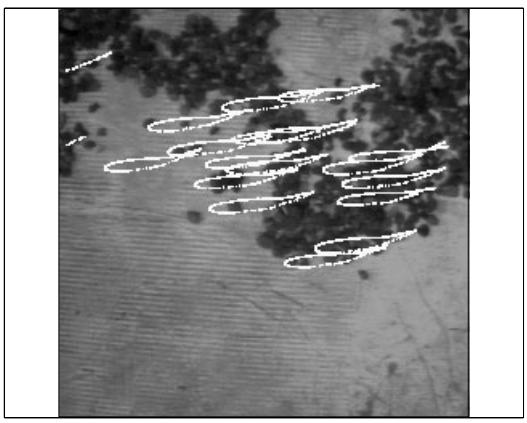
(b) Image 6



(c) Image 12

(d) Image 32

(e) Image 52



(f) Image 72

Figure 7.9: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: selected snapshots.

In the second experiment, the amplitude of the sea current was increased and set to $\mathbf{V}_c = [u_c, v_c]^T = [-0.3, -0.1]^T$ m/s. As in the previous experiment, the same behaviour with steady-state positioning errors of 1.0 cm in X and 0.4 cm in Y, with a stable null velocity is apparent (see figures 7.10 and 7.11). Compared to the previous experiment, the positioning steady-state errors increased. Indeed, there were more tracking errors between the desired and the final features' locations (see figure 7.13). This is most probably due to the fact that the deviation of the robot's position along the X axis was more important than in the previous experiment. It is clearer if the errors in features' co-ordinates of figure 7.6 and figure 7.13 are compared: the top tracks of the latter (figure 7.13) are more spread out once the robot has stabilised itself than in the former experiment (figure 7.6).

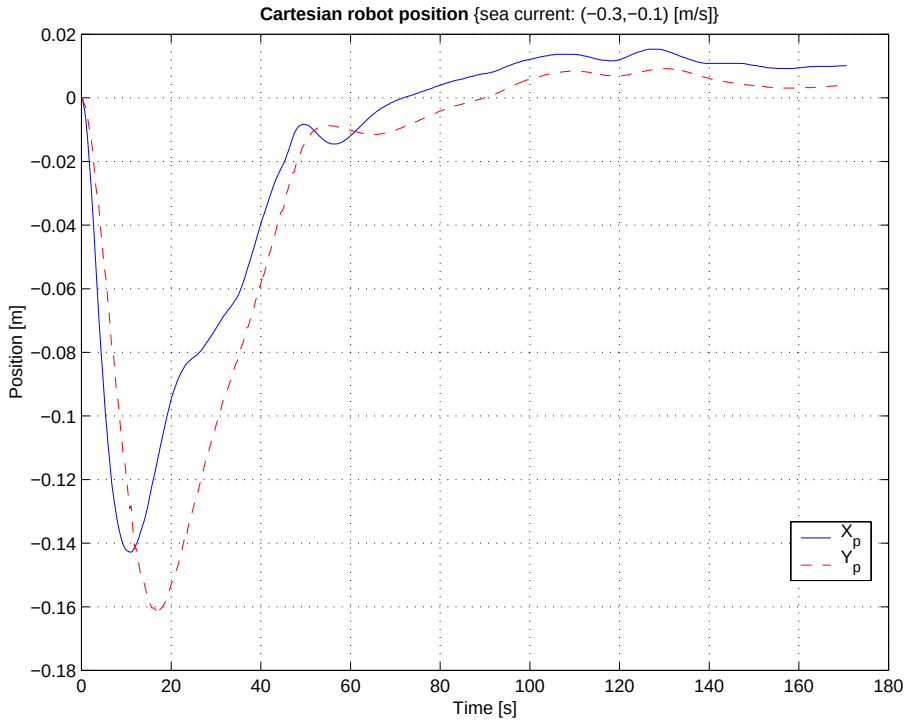


Figure 7.10: Experimental results with sea current $\mathbf{V}_c = [-0.3, -0.1]^T$: Cartesian robot's position.

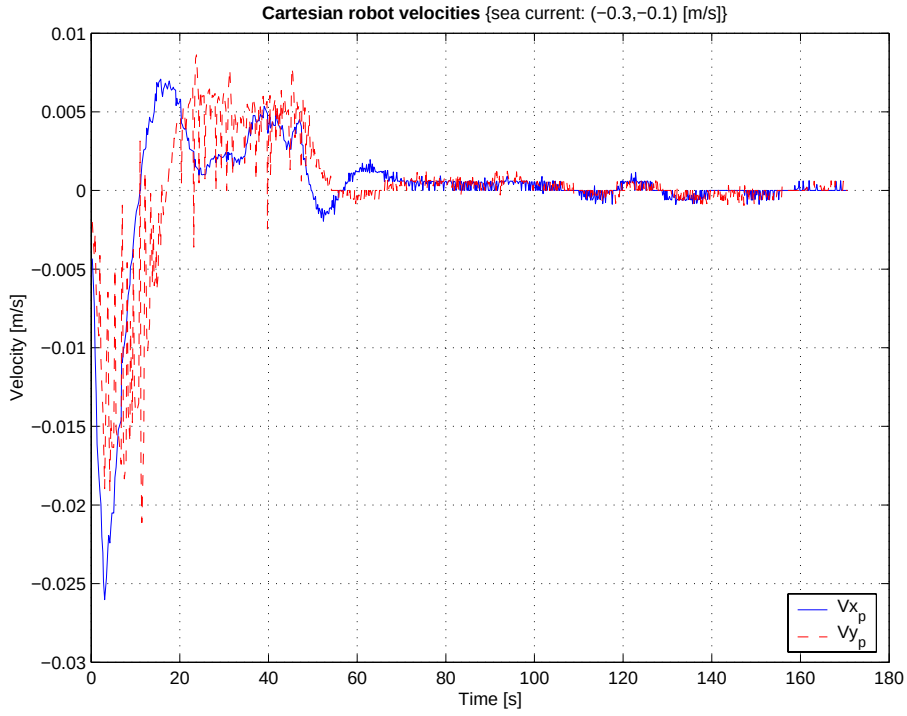


Figure 7.11: Experimental results with sea current $\mathbf{V}_c = [-0.3, -0.1]^T$: Cartesian robot's velocities.

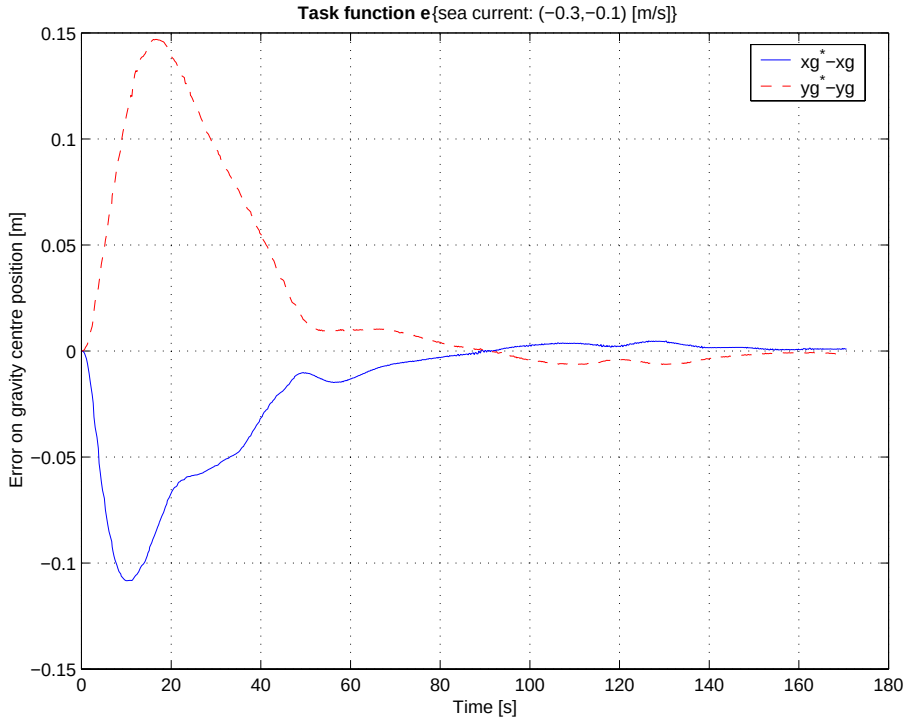


Figure 7.12: Experimental results with sea current $\mathbf{V}_c = [-0.3, -0.1]^T$: task function.

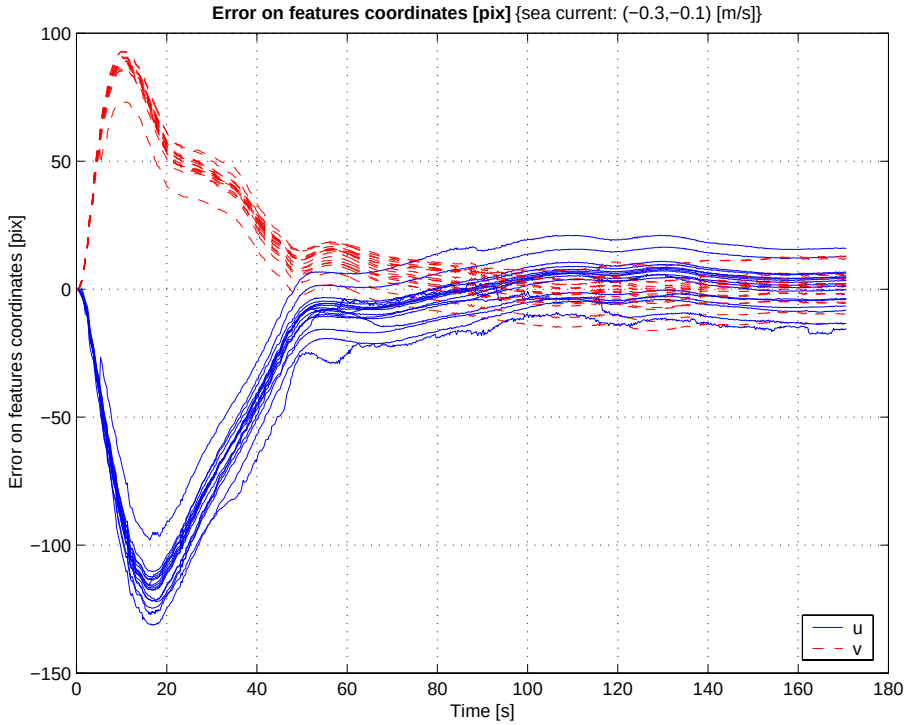


Figure 7.13: Experimental results with sea current $\mathbf{V}_c = [-0.3, -0.1]^T$: error on features' co-ordinates.

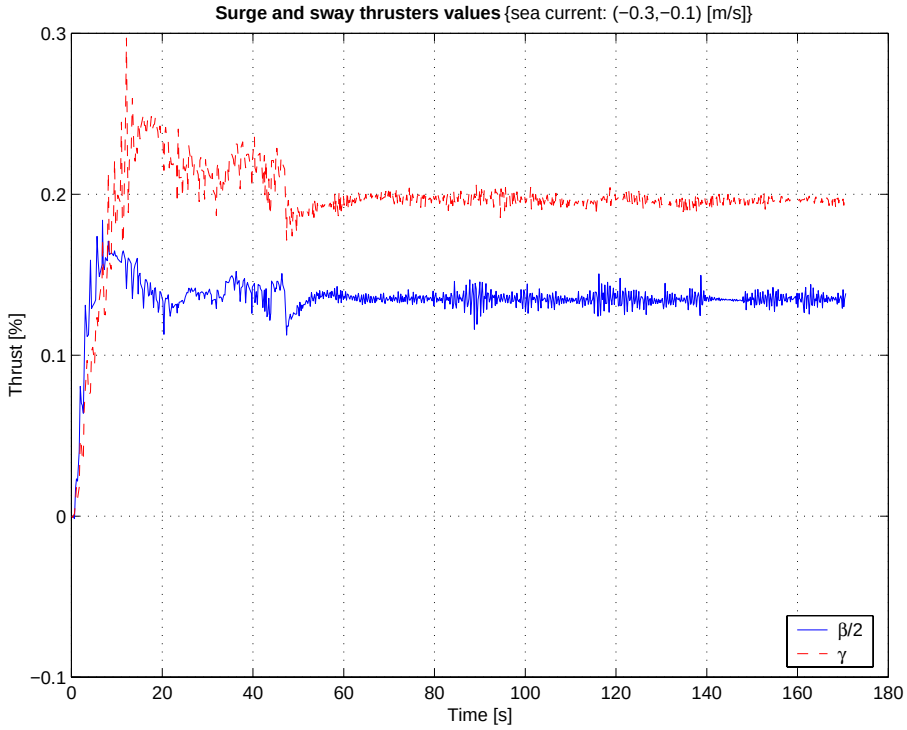


Figure 7.14: Experimental results with sea current $\mathbf{V}_c = [-0.3, -0.1]^T$: surge and sway thrusters.

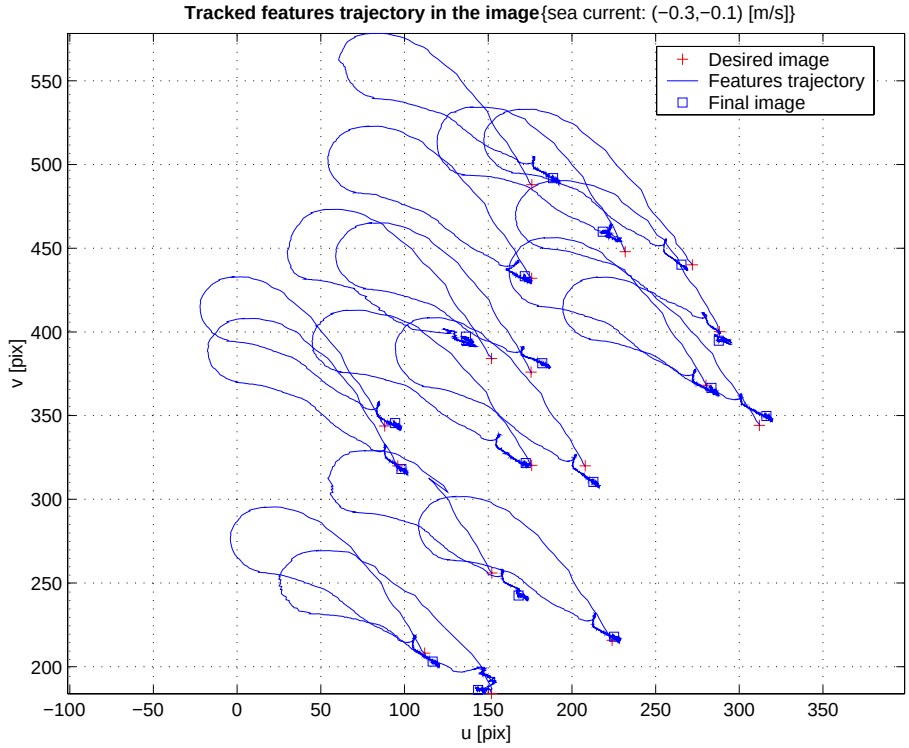
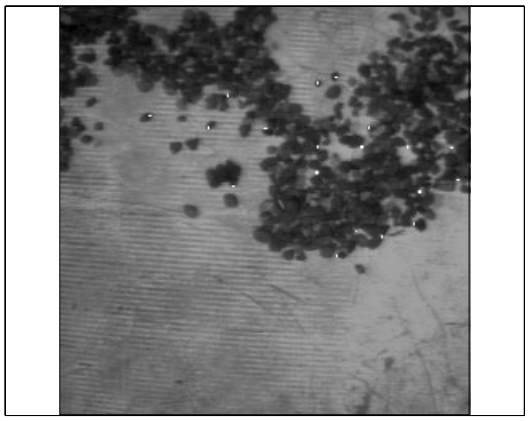
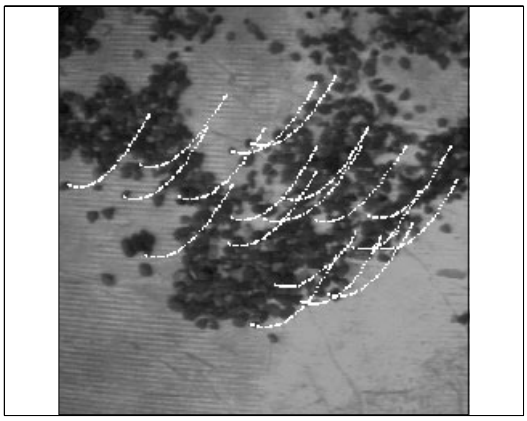


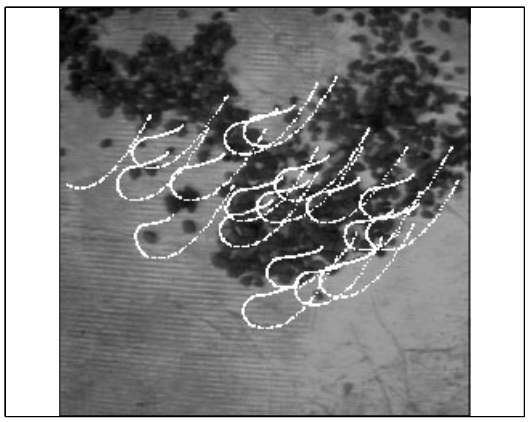
Figure 7.15: Experimental results with sea current $\mathbf{V}_c = [-0.3, -0.1]^T$: trajectory of the features in the image.



(a) Image 0

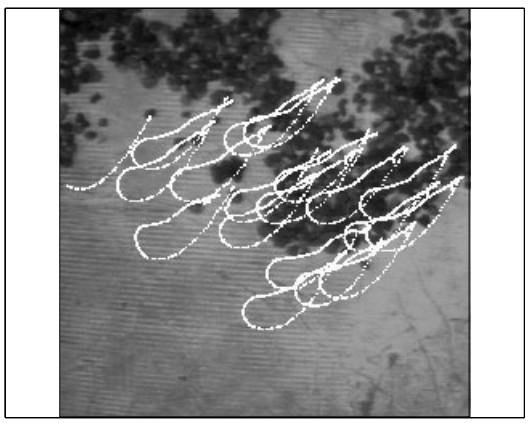


(b) Image 10



(c) Image 30

(d) Image 60



(e) Image 100

(f) Image 145

Figure 7.16: Experimental results with sea current $\mathbf{V}_c = [-0.3, -0.1]^T$: selected snaphots.

The third experiment increased the action of the current along the sway d.o.f. of the robot. Since the sway thruster provided less power than both the surge thrusters and as the transverse section of ANGUS was greater than the front section, the sea current amplitudes ANGUS could counteract in this direction were smaller. The maximum sea current's amplitude was indeed 0.2 m/s in sway compared to 0.5 m/s in surge (see next experiment). Above 0.2 m/s, the vision-controlled system was unable to meet the station-keeping objective. The general comments made in the previous experiments are still applicable. In this instance, note the degradation of the Y positioning accuracy where the steady-state error reached almost 5 cm (figure 7.17). This degradation was mostly due, as in the previous experiments, to the integration of estimation errors of features' interframe displacements. Figures 7.20 and 7.22 indeed show that one of the feature exhibited at least a 50 pixels error in horizontal co-ordinate, and globally the final values of the features' positioning errors were noticeably spread about zero.

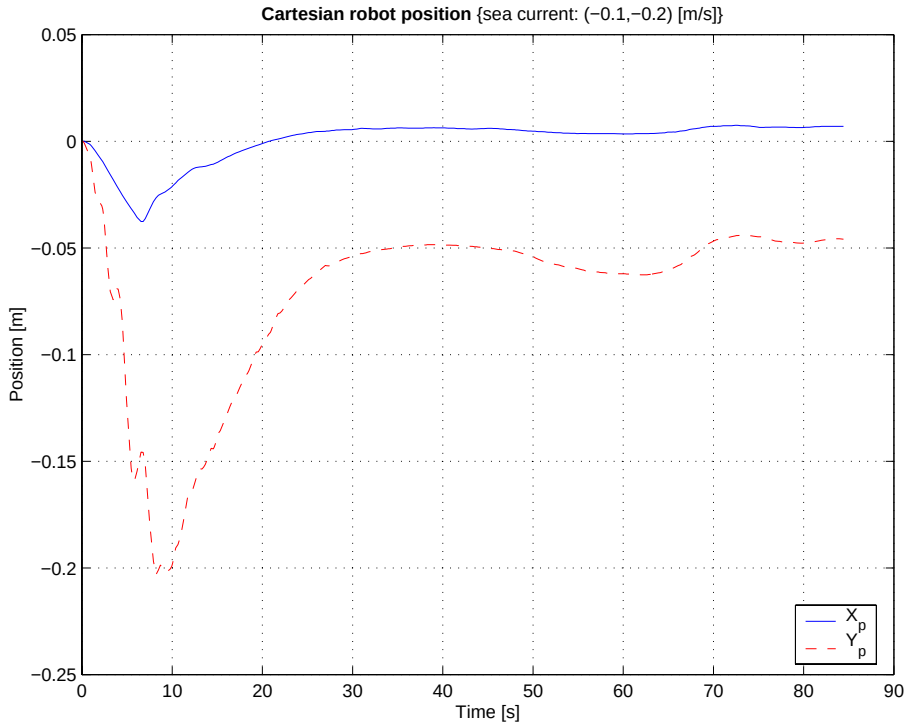


Figure 7.17: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.2]^T$: Cartesian robot's position.

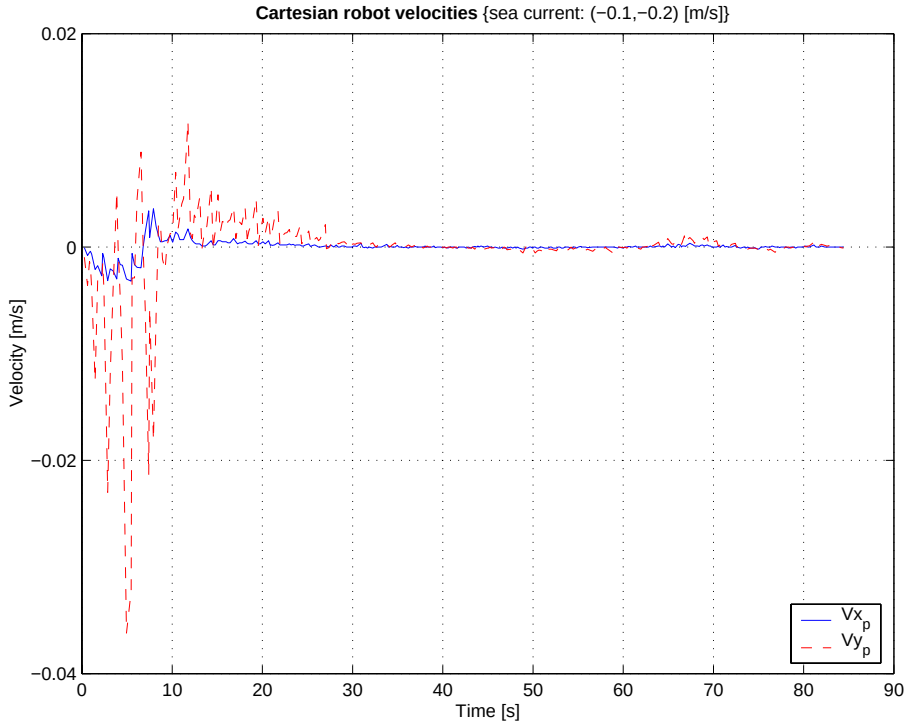


Figure 7.18: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.2]^T$: Cartesian robot's velocities.

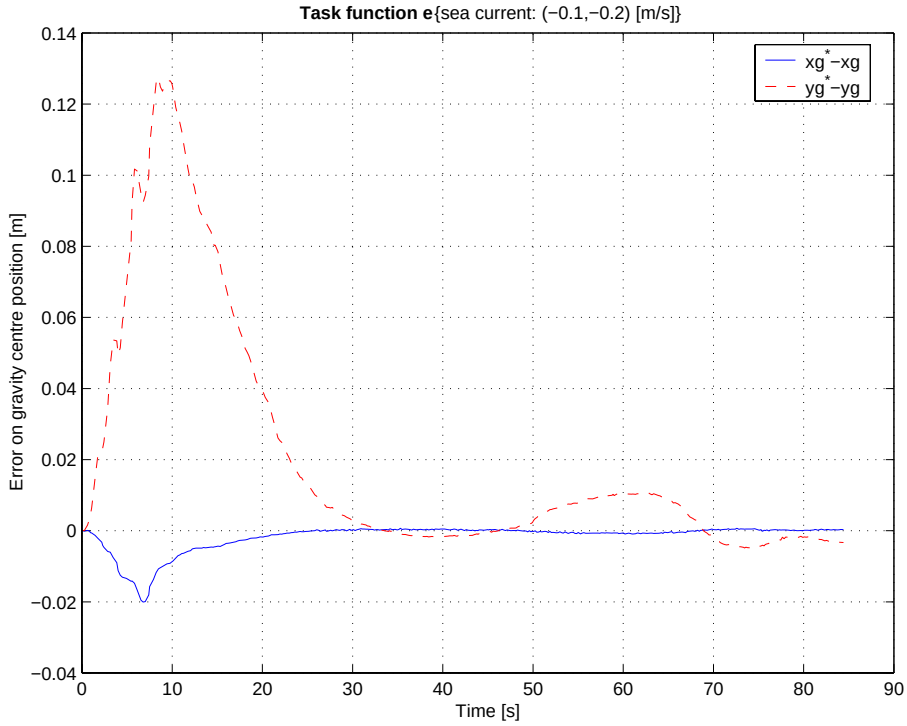


Figure 7.19: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.2]^T$: task function.

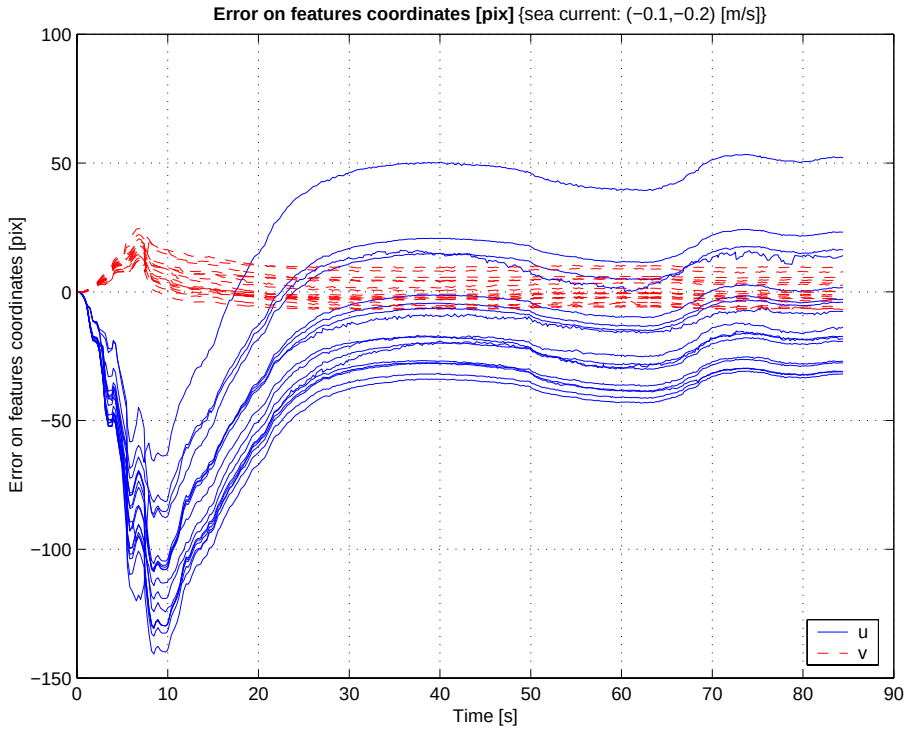


Figure 7.20: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.2]^T$: error on features' co-ordinates.

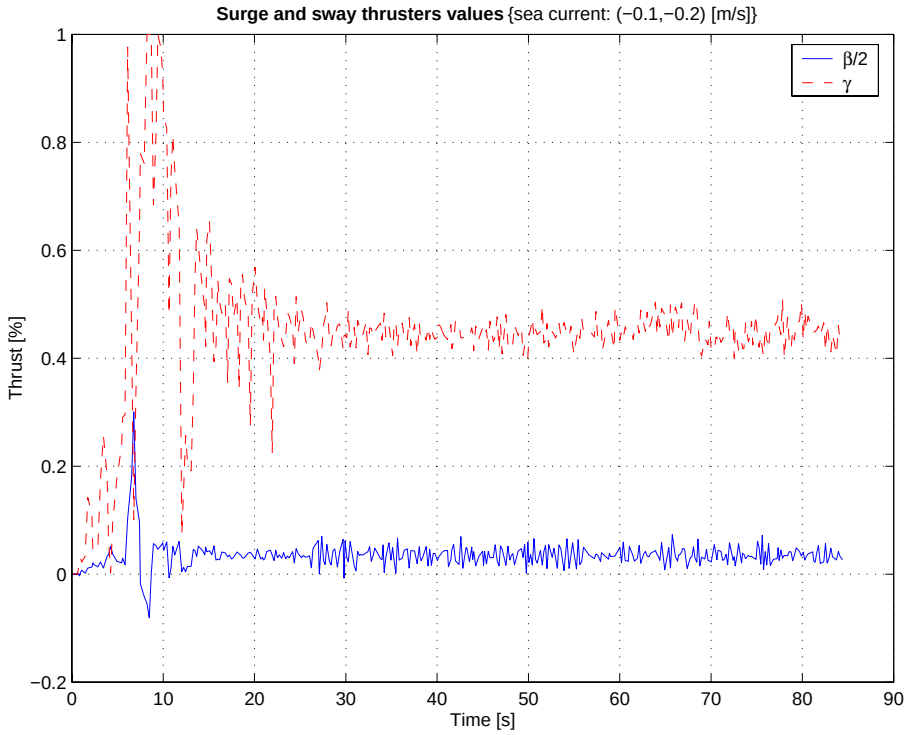


Figure 7.21: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.1]^T$: surge and sway thrusters.

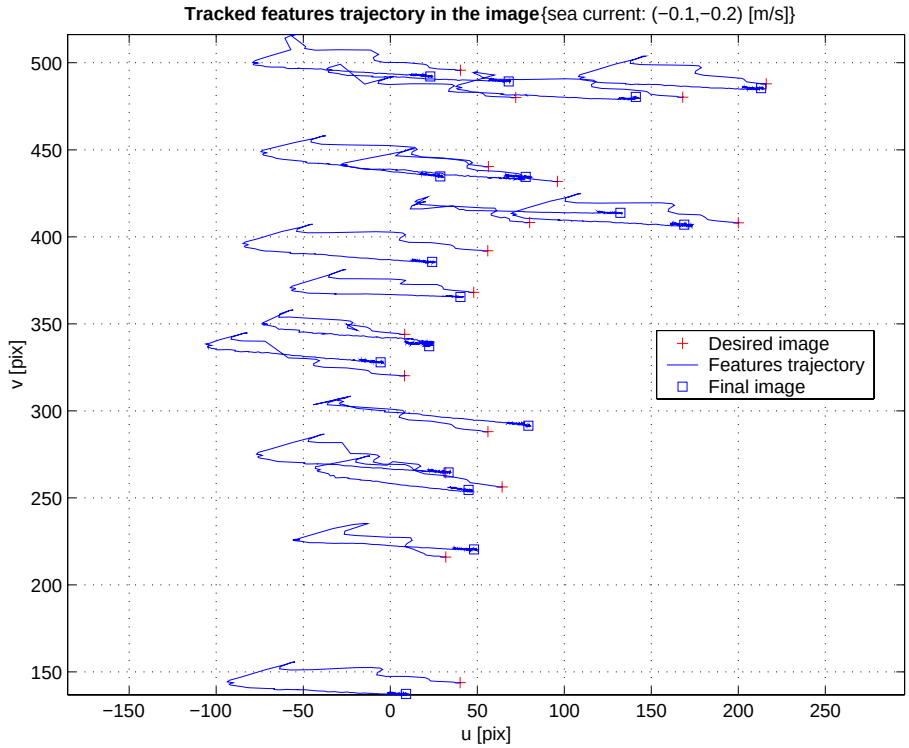
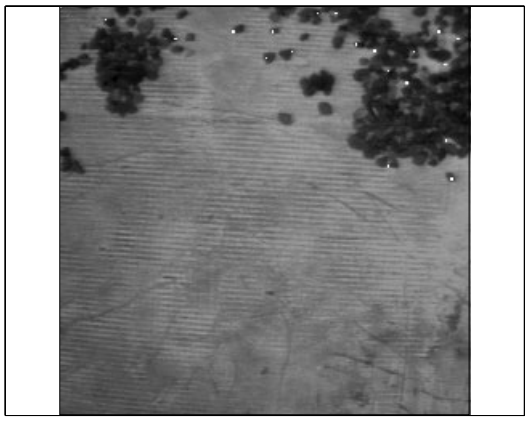
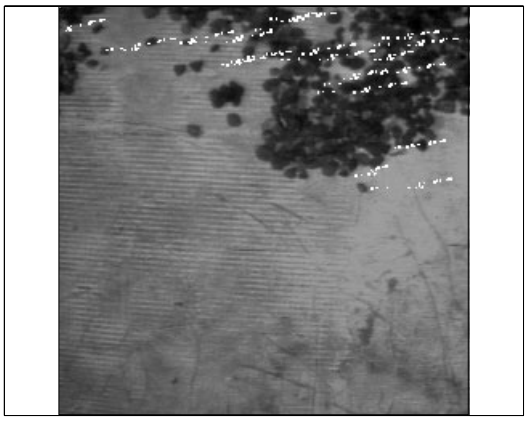


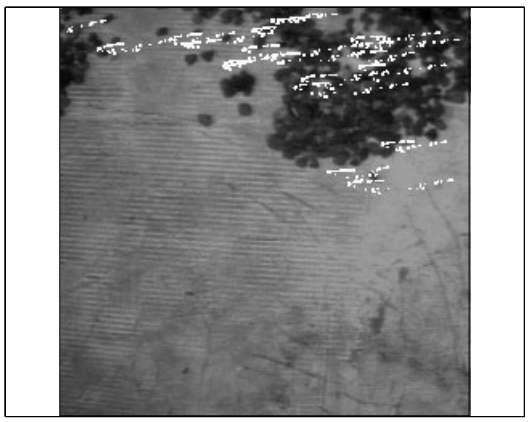
Figure 7.22: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.2]^T$: trajectory of the features in the image.



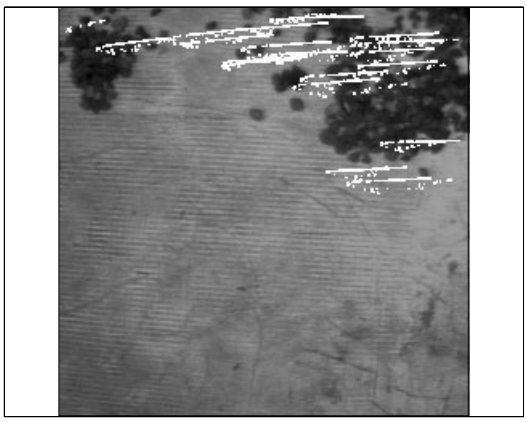
(a) Image 0



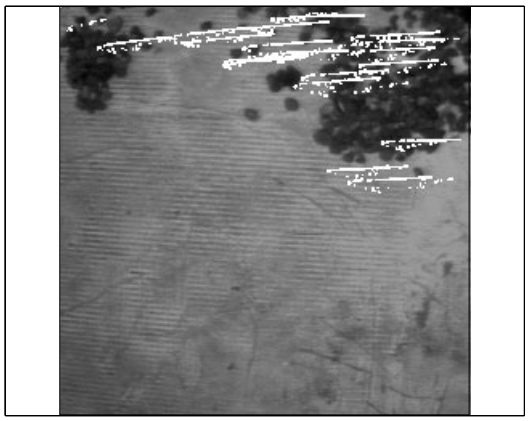
(b) Image 5



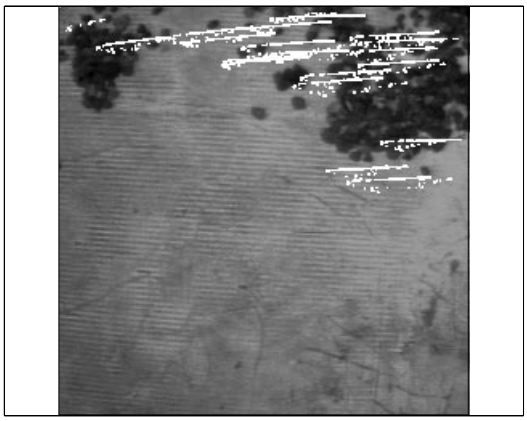
(c) Image 12



(d) Image 33



(e) Image 44



(f) Image 65

Figure 7.23: Experimental results with sea current $\mathbf{V}_c = [-0.1, -0.2]^T$: selected snapshots.

The last experiment introduced the strongest sea current that the closed-loop system could counteract along the surge axis. For this experiment, the sea current was therefore set to $\mathbf{V}_c = [-0.5, -0.1]^T$ m/s. The station-keeping was, yet again, successful with steady-state positioning errors of 3.5 cm in X and 2.2 cm in Y (see figure 7.24). The behaviour was similar than in the previous experiments. Figures 7.24 to 7.30 present the results of this experiment.

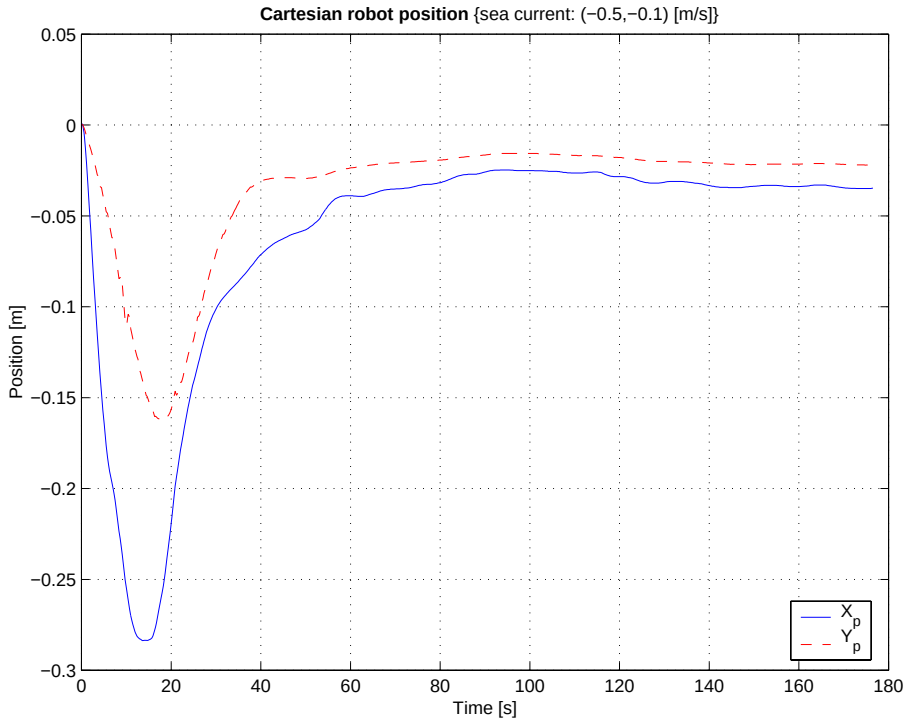


Figure 7.24: Experimental results with sea current $\mathbf{V}_c = [-0.5, -0.1]^T$: Cartesian robot's position.

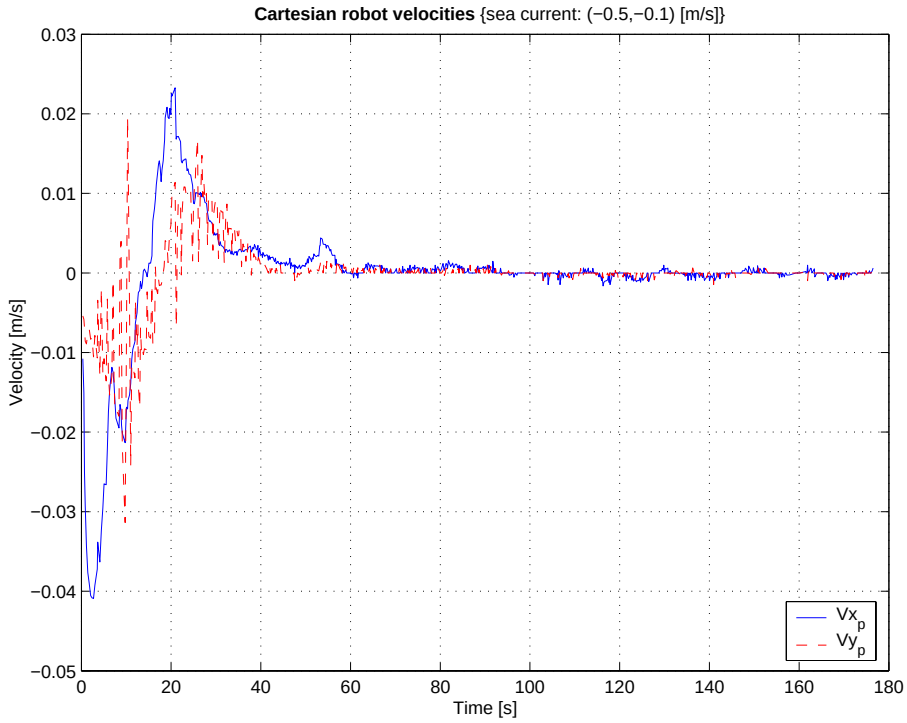


Figure 7.25: Experimental results with sea current $\mathbf{V}_c = [-0.5, -0.1]^T$: Cartesian robot's velocities.

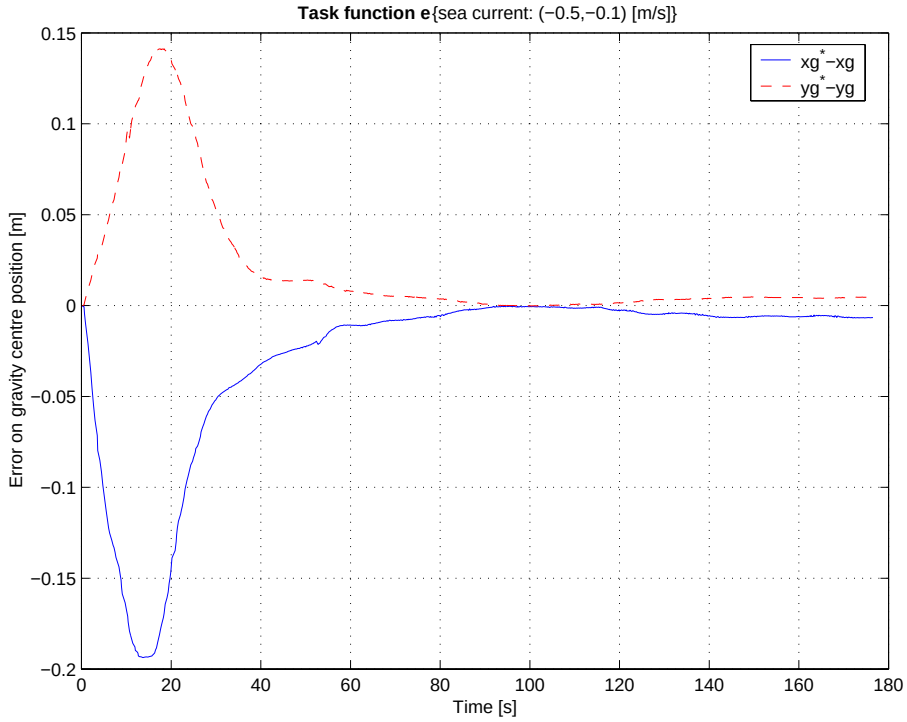


Figure 7.26: Experimental results with sea current $\mathbf{V}_c = [-0.5, -0.1]^T$: task function.

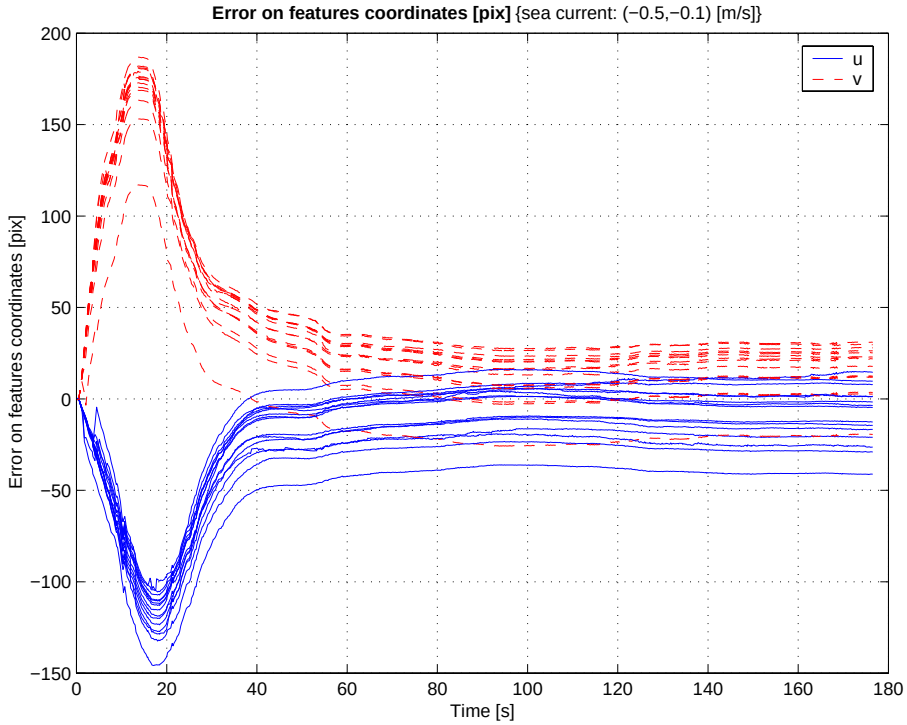


Figure 7.27: Experimental results with sea current $\mathbf{V}_c = [-0.5, -0.1]^T$: error on features' co-ordinates.

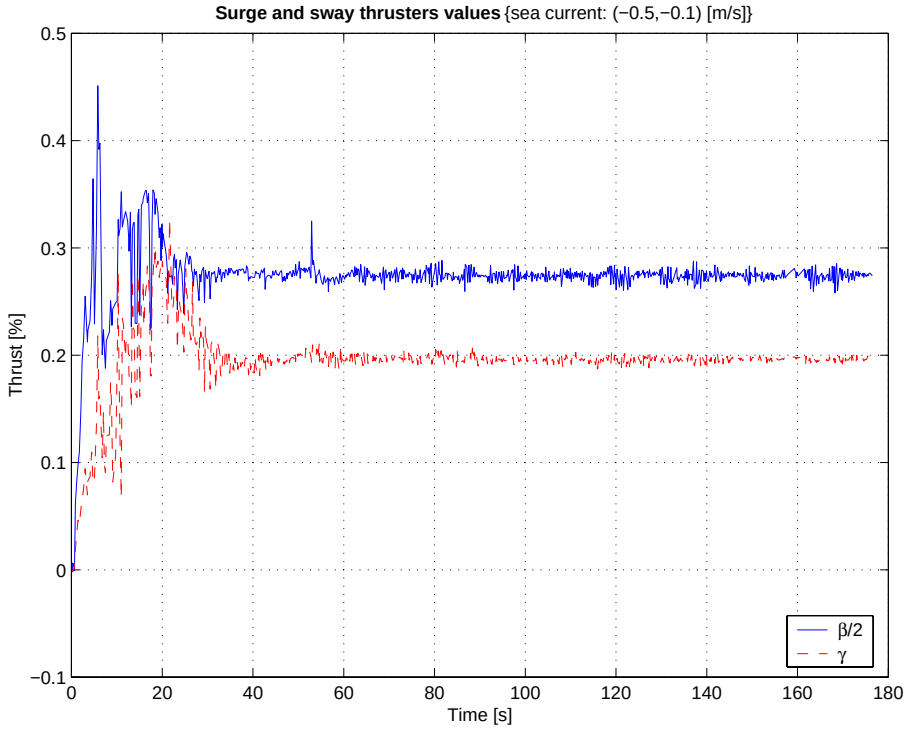


Figure 7.28: Experimental results with sea current $\mathbf{V}_c = [-0.5, -0.1]^T$: surge and sway thrusters.

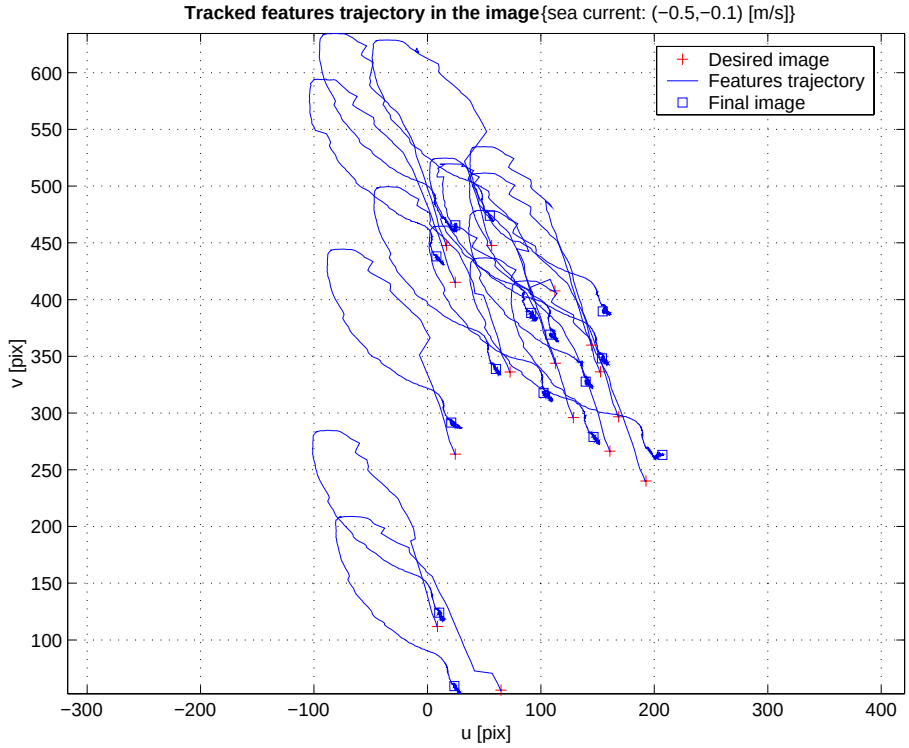
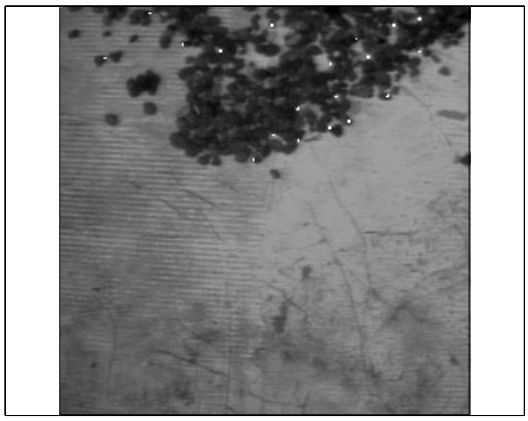
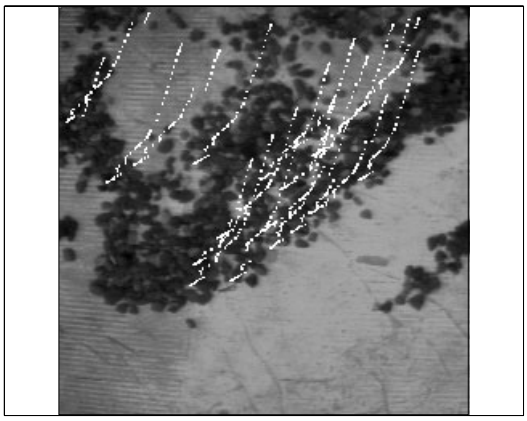


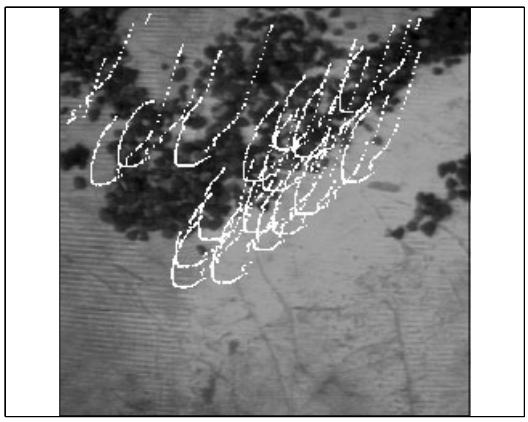
Figure 7.29: Experimental results with sea current $\mathbf{V}_c = [-0.5, -0.1]^T$: trajectory of the features in the image.



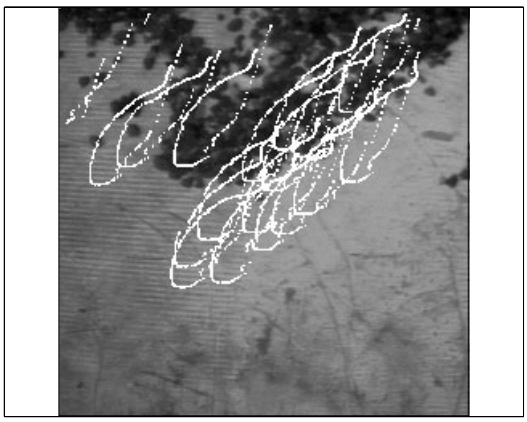
(a) Image 0



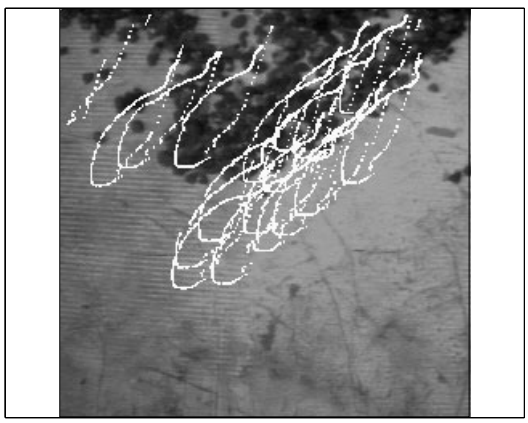
(b) Image 10



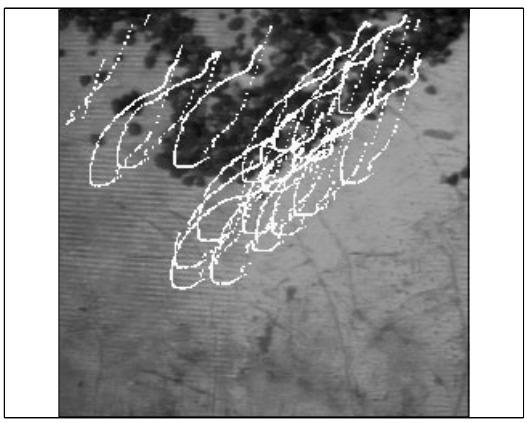
(c) Image 20



(d) Image 50



(e) Image 100



(f) Image 140

Figure 7.30: Experimental results with sea current $\mathbf{V}_c = [-0.5, -0.1]^T$: selected snapshots.

To conclude on this series of "nominal" i.e. successful visual station-keeping experiments, it can be stated that the proposed scheme proved robust to a variety of sea current amplitudes, as high as 0.5 m/s in surge and 0.2 m/s in sway. Drift in the feature tracking algorithm explained the observed steady-state positioning errors which reached a maximum value of 5 cm.

Most experiments that ran with the aforementioned range of sea current amplitudes were successful. However, the control scheme occasionally failed when too many features were lost. An example of such a failure will now be given and analysed.

7.3.3 Abnormal behaviour

Obvious failures of the visual servoing scheme occurred when all features[3] went out of the field of view. Features moved out of the field of view in two major cases. In the first one, the sea current was so strong that the robot could not cancel its action and eventually no common region between the scene in the first image and the scene in the current image remained. In the second case, the sea current's action could be canceled but the features were extracted mostly from either side of the first image and moved out of the field of view before the robot could come back towards its initial position. The experiment presented here fitted in the second case.

The sea current was $\mathbf{V}_c = [0.0, -0.2]^T$ m/s, i.e. no disturbance was applied along the surge axis. The main consequence of this choice is that the distance required for the features to leave the image was considerably small since the motion remained along the sway d.o.f. This statement is illustrated more clearly by the snapshots of the underwater scene of figure 7.37.

This experiment's results were, until time instant $t = 146.8$ seconds, similar to the so-called "nominal" experiments of the previous section. At that time instant, one of the last four remaining features was lost and the homography could not be computed anymore, and the servoing was stopped.

Although similar, a few other differences were observed, especially on the thrusters' plot of figure 7.35 where the sway thruster signal exhibited several discontinuities

[3]Or all but three at least to be more precise.

(see the peaks). The most probable reason is that this was caused by the successive loss of features during the servoing process combined with increasing errors on the estimation of the remaining features' locations. This can be easily explained. Assume that there were only four features tracked during the experiment. To build the task function, the co-ordinates of the centre of gravity of the four features in the first image of co-ordinates (x_g^*, y_g^*) were computed, and were compared with the current co-ordinates (x_g, y_g). If one of the features was grossly misplaced, the current location of the centre of gravity would be biased toward that feature point compared to the "true" location of the centre of gravity if the features were perfectly tracked. If that happened, a discontinuity in the task function would occur. Such a discontinuity explained the "peaks" of the sway thruster signal (figure 7.35) as well as the sudden changes of direction of the robot's velocity along the sway axis (figure 7.32).

In figure 7.34, at least four important "jumps" in the first twenty seconds are noticeable on the remaining tracked features' displacements. This can be explained simply. Indeed, it was seen in chapter 5 that the feature tracker was performing an approximate local correlation within a search window to track features in a new image. Since the features were extracted on pieces of gravel, it was very likely that one piece of gravel and another close one presented almost the same characteristics at one time instant. If, in addition, the latter also had a higher correlation score due to slight lighting variations, it would then be chosen (wrongly) as the tracked feature to the detriment of the true feature.

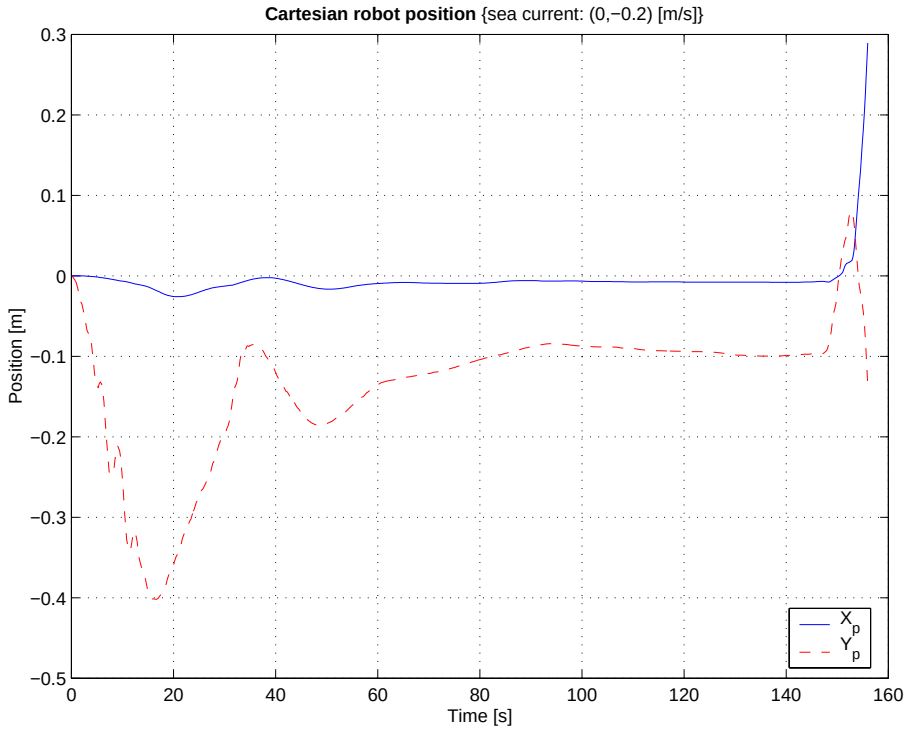


Figure 7.31: Experimental results with sea current $\mathbf{V}_c = [0, -0.2]^T$: Cartesian robot's position.

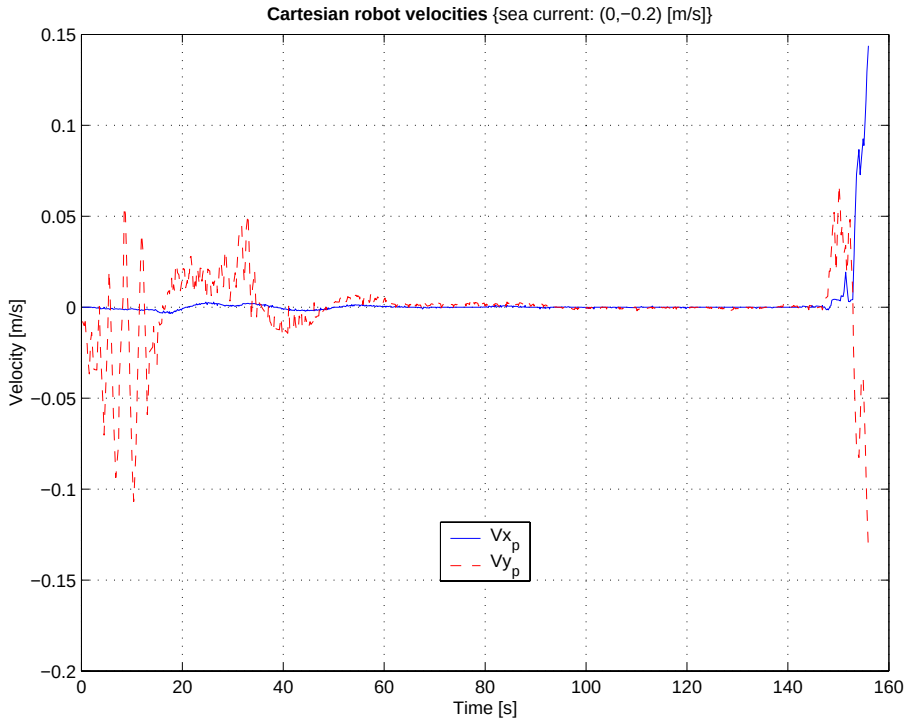


Figure 7.32: Experimental results with sea current $\mathbf{V}_c = [0, -0.2]^T$: Cartesian robot's velocities.

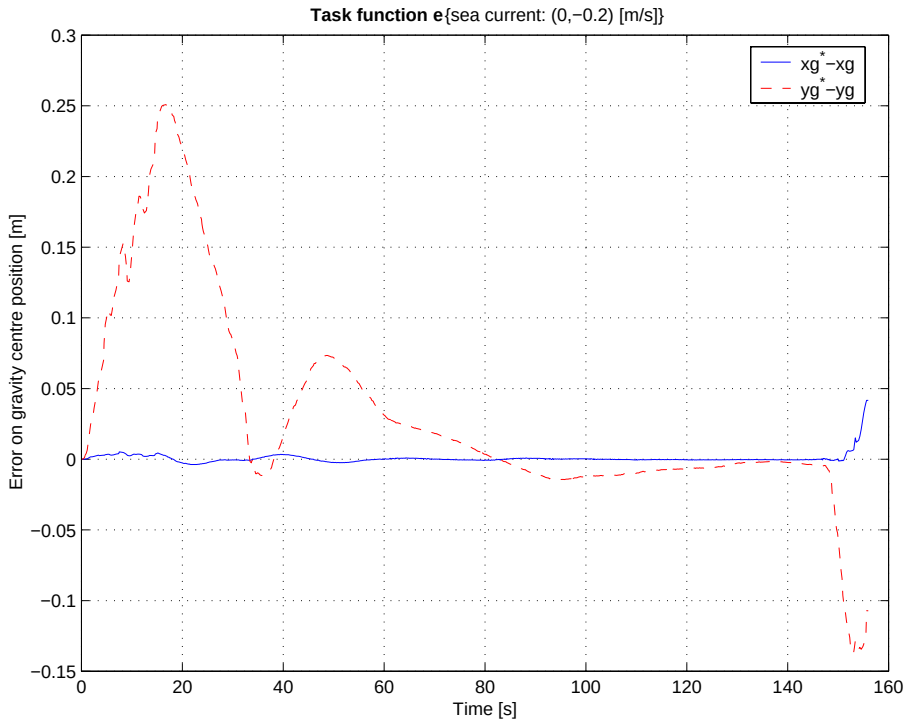


Figure 7.33: Experimental results with sea current $\mathbf{V}_c = [0, -0.2]^T$: task function.

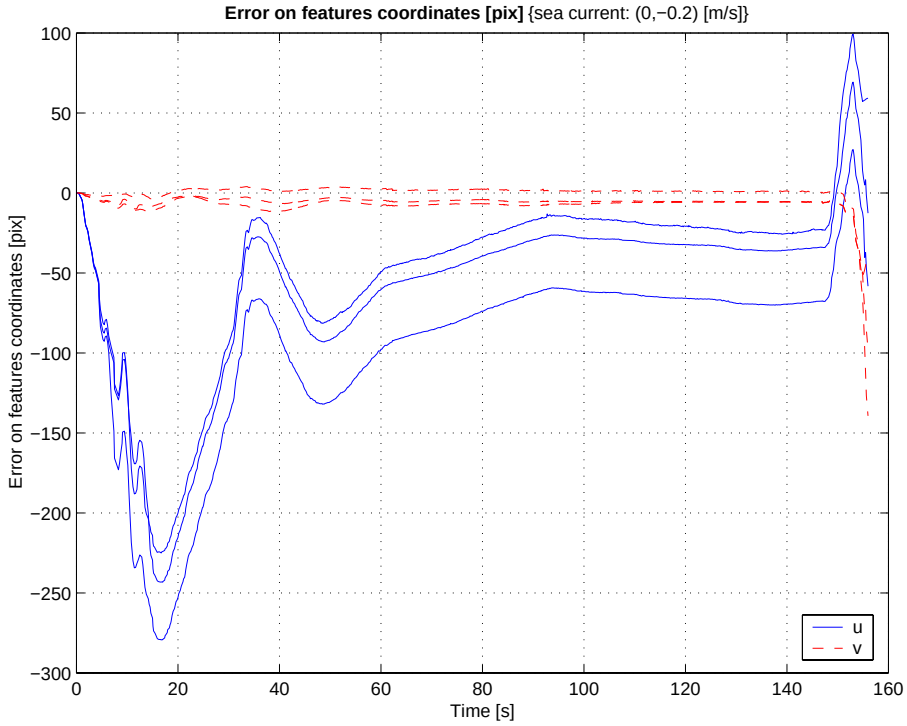


Figure 7.34: Experimental results with sea current $\mathbf{V}_c = [0, -0.2]^T$: error on features' co-ordinates.

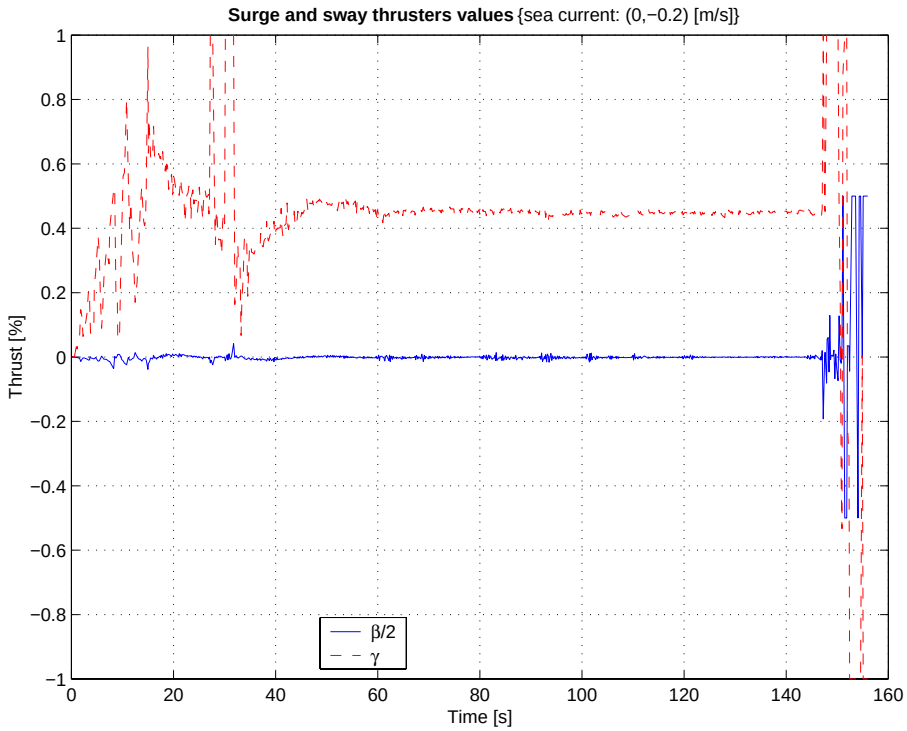


Figure 7.35: Experimental results with sea current $\mathbf{V}_c = [0, -0.2]^T$: surge and sway thrusters.

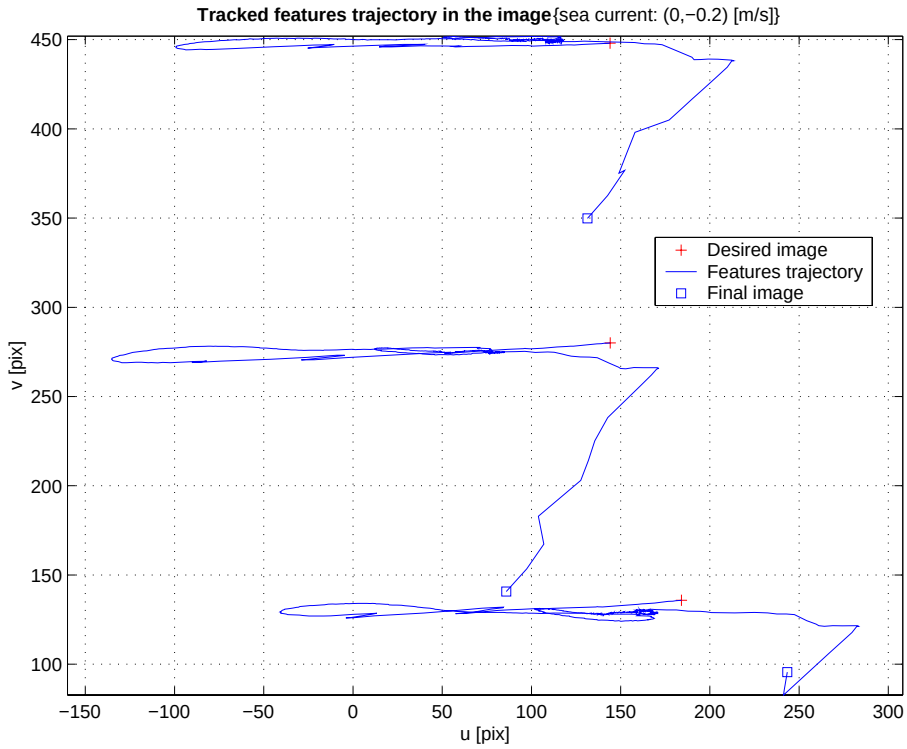
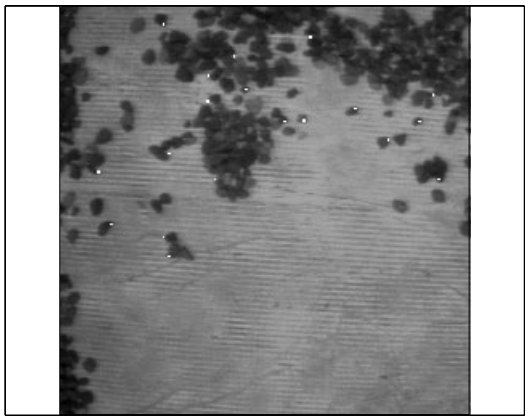
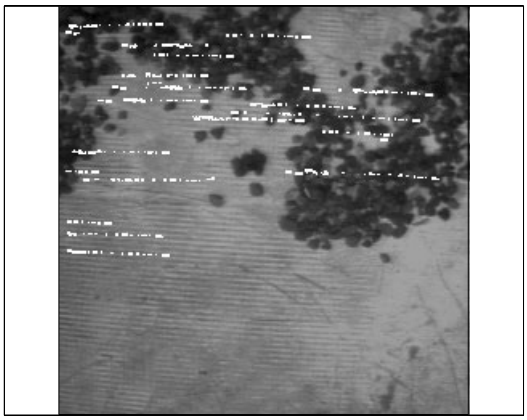


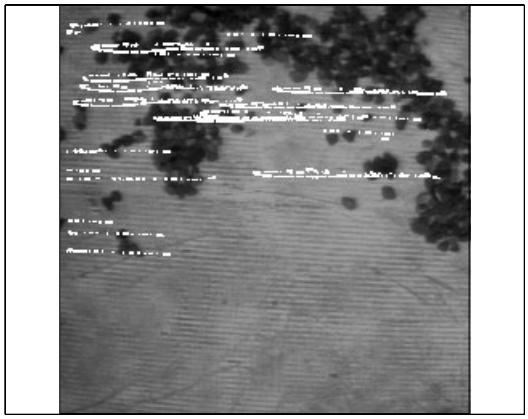
Figure 7.36: Experimental results with sea current $\mathbf{V}_c = [0, -0.2]^T$: trajectory of the features in the image



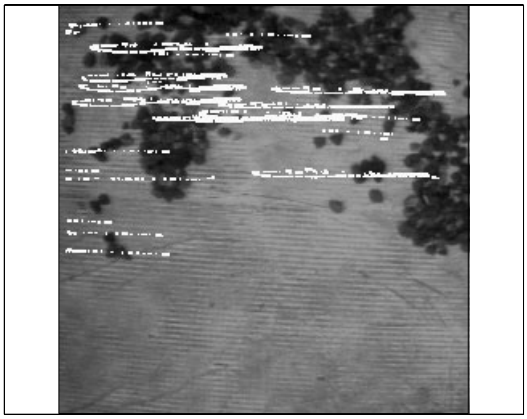
(a) Image 0



(b) Image 10

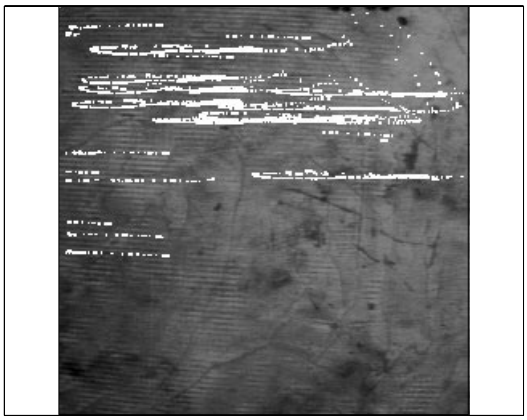


(c) Image 30



(d) Image 60

(e) Image 90



(f) Image 123

Figure 7.37: Experimental results with sea current $\mathbf{V}_c = [0, -0.2]^T$: selected snapshots.

7.3.4 Discussion

The results clearly demonstrated that the stabilisation of the robot was possible. Figures 7.4, 7.11, 7.25, and 7.18, illustrated the asymptotic convergence of the Cartesian robot's velocity to zero.

However, slight steady-state positioning errors were present (figures 7.3, 7.10, 7.24, and 7.17). This behaviour was explained by the drifting of the tracked features in the images (see chapter 5). Indeed, if a given feature was drifting from its original location, the position of the computed centre of gravity of the whole set of features was biased. In other words, the task function $\mathbf{e}$ reached zero as required (figures 7.5, 7.12, 7.26, and 7.19), but its minimum did not correspond anymore to the initial position of the robot. However, the positioning errors remained smaller than 5 cm in all "nominal" cases.

The features' drift was apparent in the plots of the features' displacements through time (figures 7.6, 7.13, 7.27). Defining as task function the centre of gravity of the set of tracked features actually made the proposed method more robust to drift by averaging its adverse contribution. Comparing for example figures 7.5 and 7.6 illustrates that point. The drift was also obvious in the snapshots of figure 7.30.

Failures in the experiments occurred when the sea current was high enough to prevent the robot to keep enough features in the field of view while trying to counteract the current's effect. The PID gains were intentionally set to achieve an overdamped behaviour, therefore the time response was rather slow.[4] Higher gains would have allowed to counteract greater currents. However, the robot's accelerations would have been more important, and, as a result, the feature tracker might have lost track. Indeed, interframe displacements of more than 4 pixels caused the tracker to fail as seen in chapter 5.

To avoid a failure such as the one illustrated by the example of section 7.3.3, a possible solution would have been to extract new features during the control process and redefine the desired centre of gravity from the new extracted features' set. Ob-

[4] In [47], an underdamped behaviour was chosen, thus providing a faster time response to the cost of some overshooting. As discussed previously in chapter 6, it is clearly a designer's choice depending on the exact requirements of the station-keeping application.

viously, changing the reference point would provoke a bias in the robot's positioning error. Indeed, once the robot had moved of 10 cm at instant $t_1 > 0$ for instance, and the feature tracking was reinitialised, the new reference point would be 10 cm away from the initial one. Therefore a systematic error would appear. This thesis did not implement reinitialisation so as to easily compare the positioning errors and because the goal of the visual station-keeping control scheme was to stay at the initial reference point. That was therefore a designer's choice. For a less strict application, i.e. staying as immobile as possible and allowing some drift, extracting features when needed would have been suitable.

7.4 Conclusions

This work showed the validity of 2-D visual servoing for underwater vehicle station keeping. This approach was robust to various sea current disturbances and achieved good positioning precisions. The use of a linear PID controller permitted slow, but very stable hover capabilities. Besides, these experiments were performed on a number of unmarked planar targets without any changes in the feature tracking parameters, which demonstrated the robustness of the visual processing part.

The performances of the proposed control scheme were characterised in terms of admissible sea current disturbances, and positioning accuracies. The effect of undesirable feature tracker behaviour on the dynamic positioning performances was analysed and explained, and in particular the effect of features' drift, features' mislocations and limited field of view.

Obviously, better transient responses could be obtained if the feature tracking algorithm was faster. In the performed experiments, the average processing time of an image was of 180 ms, which overall limited the visual control loop to a frequency of 5 Hz. Disturbance rejection could also be improved by implementing a multi-scale feature tracking algorithm, thus allowing the robot to further accelerate without causing the visual processing to break down. As discussed in chapter 5, a trade-off must be made between positioning accuracy and processing speed.

Finally, incorporating nonlinear controllers, that use the vehicle's dynamics knowl-

edge, as described in [64], would also improve the disturbance rejection if the demanded accelerations met the feature tracker's limits.

Chapter 8

Conclusions

8.1 Summary

Unmanned underwater vehicles lack a good ability to hover with conventional underwater sensors. These sensors suffer from several drawbacks such as low sampling rates, low resolution, complexity of operation, drift and cost, as seen in the introduction. This makes the control design challenging. Underwater video cameras, however, can provide position information with respect to a local object — a rock on the seabed for example — with a higher frequency and a better resolution than most conventional underwater sensors. Therefore, it was natural to consider introducing a camera into the control loop of underwater vehicles. The main objective of this thesis was to investigate methods, *based on visual information alone*, to dynamically position a vehicle with respect to a fixed object. These methods also had to have computing power's needs compatible with standard off-the-shelf embedded computers suitable for Autonomous Underwater Vehicles.

Visual control of robots is a multi-disciplinary field at the cross-road of topics such as robot modelling and control, computer vision and image processing. It was then necessary to investigate and introduce the basic concepts required in visual servoing methods.

Chapter 2 provided an introduction to underwater vehicle modelling and control. It highlighted the peculiarities of mobile robots travelling in water and the control issues encountered by underwater vehicles.

The ability to measure the location of an underwater vehicle is required to evaluate the performances of dynamic positioning. Unfortunately, conventional on-board sensors could not provide this information, and even acoustic transponders could not yield enough accuracy. Emulating the dynamic behaviour of a typical ROV

on a Cartesian robot in a water tank solved this issue. Methods to replicate the behaviour of the ANGUS ROV were investigated and developed (chapter 3). The Cartesian robot was able to replicate within centimetric accuracy the trajectory of the two degrees-of-freedom model of ANGUS. It was however not possible to emulate exactly the velocity profile of the dynamic model with the Cartesian robot. However, the behaviour of the compounded system: Cartesian robot and underwater vehicle model, was still representative of a typical, but slightly slower than ANGUS, ROV. The Cartesian robot allowed to get positioning measurements in the horizontal plane to within a few millimetres. To the author's knowledge, it was the first time that such a method was used to evaluate the positioning accuracy of visual control techniques for underwater vehicles.

A prerequisite of visual servoing techniques is the ability to infer motion information from images. This is one of the issues of computer vision research. Chapter 4 gave an account of linear motion estimation techniques based on the epipolar geometry. To obtain metric information on the camera motion, it is necessary to know the intrinsic parameters of the camera. Estimating these parameters is the aim of camera calibration. It requires 3-D patterns whose geometry is known and is generally performed off-line, prior to operation. Recently, self-calibration techniques which allow an automatic and on-line estimation of camera parameters have been proposed. A review of the main methods was also given in this chapter. These methods could be of use since the intrinsic parameters of a camera are not always constant. Mechanical and thermal variations, which are very likely to happen on an underwater camera during UUV operation, modify the intrinsic parameters. Zooming will also change the parameters.

Most computer vision and visual servoing algorithms call for point correspondences between two views. The ability to track features throughout time is therefore crucial. The thesis' contribution lay in the experimental evaluation of a feature tracker in underwater conditions. In particular, the tracker's sensitivity with respect to illumination, the maximum interframe displacement allowed and the consistency of tracking were investigated in a water test tank (chapter 5).

The issue of defining the visual servoing task was investigated. In particular, a visual

control scheme was proposed to station-keep the six degrees-of-freedom model of ANGUS. This scheme, adapted from the 2 1/2 D visual servoing algorithm [50], took into account the underactuation of ANGUS (roll and pitch were not controllable). The visual servoing task was therefore physically achievable by the vehicle. The proposed algorithm was demonstrated to be robust to sea current disturbances. The effect on noise on the feature tracking was also assessed. In addition, the consequence that an inclined visual target would have on the visual servoing scheme was also investigated. The visual control scheme proved also to be robust with respect to the orientation of the visual target. The Cartesian robot did not possess enough degrees-of-freedom to completely emulate ANGUS. Therefore those assessments were carried out in simulation (chapter 6). The simulations demonstrated the feasibility of visual dynamic positioning for UUVs. It did not however allow the evaluation of the effect of real-time feature tracking and robot control.

In order to investigate these aspects, a two degrees-of-freedom version of a typical ROV model was emulated on the Cartesian robot (as in chapter 2). An underwater camera was attached to the robot and features were tracked at the bottom of the water tank while the ROV model was subject to sea current disturbances. Another visual servoing task was then proposed to station-keep the robot with respect to unmarked natural objects on the sea floor (chapter 7). The experimental setup allowed to assess the positioning accuracy of the proposed method. In particular, it was demonstrated that the visual servoing scheme was robust to sea current disturbances, accurate to within a few centimetres, and stable. The limitations of the proposed visual scheme were also pointed out and their cause was analysed.

8.2 Contributions

To sum up, the main achievements of this thesis were:

- the emulation of a typical underwater vehicle's behaviour on a Cartesian robot which provided ground truth positioning measurements;
- an experimental evaluation underwater of the feature point tracker properties:

sensitivity to illumination, admissible interframe displacements and tracking consistency;

- the design of a hybrid visual servoing scheme adapted to an underactuated ROV, and its performance assessment, in simulation, with respect to noise in the feature tracking, sea current disturbances and target's orientation;

- and finally, the design of a 2-D visual servoing method for the dynamic positioning of a 2 d.o.f. underwater vehicle model whose behaviour was replicated on a Cartesian robot underwater. The feature tracker assessed previously provided point matches. The positioning performance of the visual station-keeping method was quantified. The proposed method was also demonstrated to be robust to sea current disturbances.

8.3 Future work

This thesis' work demonstrated that the dynamic positioning of a typical underwater vehicle with visual servoing was possible in a wide range of situations. Limited hardware did not permit to evaluate experimentally the proposed algorithms for all six d.o.f. Transfering the visual servoing techniques to an underwater vehicle should clearly be the next step, and would allow sea trials.

In this thesis, simple linear PID controllers were employed to perform the visual servoing task. No a-priori knowledge of the underwater dynamics of the vehicle was used to enhance the control performances. On a "real" underwater robot, if such knowledge is available, it could be used. More sophisticated nonlinear control techniques could also be employed to improve transient responses and disturbances rejection.

The feature tracking was carried out on full resolution 512×512 images. As a result, the admissible interframe displacement was limited to 4 pixels. To allow greater vehicle velocity and be able to reject higher disturbances, multi-resolution techniques should be investigated. This should have the advantage of providing features at a faster rate, therefore increasing the robot's control bandwidth. The

tradeoff will obviously reside in positioning accuracy.

Appendix A

ANGUS dynamic model numerical values

The numerical values of the dynamic parameters of ANGUS 003 used in this thesis are listed below.

$$\mathbf{M} = \begin{bmatrix} 1800 & 0 & 0 & 0 & 50 & 0 \\ 0 & 2200 & 0 & -35 & 0 & 110 \\ 80 & 0 & 3200 & 0 & -50 & 0 \\ 0 & -30 & 0 & 600 & 0 & 15 \\ 50 & 0 & -50 & 0 & 850 & 0 \\ 0 & 110 & 0 & 15 & 0 & 750 \end{bmatrix}$$

$$\mathbf{B}(\boldsymbol{\nu}) = \begin{bmatrix} -350 & 0 & 0 & 0 & 35 & 0 \\ 0 & -680 & 0 & -60 & 0 & 160 \\ 30\,\frac{u}{|u|} & 0 & -1300 & 0 & -33 & 0 \\ 0 & -60 & 0 & -300 & 0 & 20 \\ 35 & 0 & -33 & 0 & -550 & 0 \\ 0 & 140 & 0 & 5 & 0 & -450 \end{bmatrix}$$

$$\mathbf{E}(\boldsymbol{\nu}) = \begin{bmatrix} 250 & 250 & 0 & 70\,u \\ 0 & 0 & 250 & 50\,v \\ 0 & 0 & 0 & 400 \\ 0 & 0 & -15 & 0 \\ 25 & 25 & 0 & 15 \\ 100 & -100 & 0 & 20\,r \end{bmatrix}$$

$$\mathbf{F}(\boldsymbol{\nu}) = \begin{bmatrix} 100 & 100 & 0 & -340\,u \\ 0 & 0 & 0 & -250\,v \\ 0 & 0 & 0 & 50 \\ 0 & 0 & 0 & 0 \\ 5 & 5 & 0 & 0 \\ 40 & -40 & 0 & -100\,r \end{bmatrix}$$

$$\mathbf{D}^T = \begin{bmatrix} 0.6 & 0.6 & 0.4 & 1.2 & 1.1 & 0.6 \end{bmatrix}$$

The buoyancy force is $W = 6031\,N$, the dry mass is $m = 615\,kg$ or a weight of $W = 6027\,N$. The metacentric height is $H = 0.165\,m$.

Appendix B

Data sheet of the underwater camera

The specifications of the underwater "Micro" camera of Mariscope Meerestechnik, used in this thesis' experiments, are extracted "as is" from their product data sheet.

Type	1/3 " CCD bw-ship
Picture elements	512 (H) × 582 (V) pixel
Resolution	380 TV lines
Shutter	ELC-automatic
Output	HF frequency modulated
Lens	4.48 mm, F:1.8
Sensitivity	0.1 lux
Viewing angle	48 ° in water

Table B.1: Specifications of the underwater camera.

References

[1] Y. Bar-Shalom and T. E. Fortmann. *Tracking and data association*, volume 179 of *Mathematics in science and engineering*. Academic Press, 1988.

[2] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[3] P. Bellec. Simulation of the Six-Degrees-of-Freedom motion of a remotely controlled unmanned submersible. Master's thesis, Department of Electrical and Electronic Engineering, Heriot-Watt University, Edinburgh, UK, 1980.

[4] M. Caccia, G. Bruzzone, and G. Veruggio. Hovering and altitude control for open-frame UUVs. In *IEEE Conference of Robotics and Automation*, volume 1, pages 72–77, 1999.

[5] F. Chaumette. *La relation vision-commande: théorie et application à des tâches robotiques*. PhD thesis, Université de Rennes, IRISA, 1990.

[6] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In G. H. D. Kriegman and A. Morse, editors, *The confluence of vision and control*, volume 237 of *LNCIS*, pages 66–78. Springer Verlag, 1998.

[7] R. Cipolla and N. Hollinghurst. Visually guided grasping in unstructured environments. *Robotics and Autonomous Systems*, 19(3-4):337–346, March 1997.

[8] P. I. Corke. *Visual control of robots: High-performance visual servoing*. Research Studies Press Ltd, Taunton, Somerset, England, 1997.

[9] P. I. Corke and S. A. Hutchinson. Real-time vision, tracking and control. In *IEEE Conference on Robotics and Automation*, volume 1, pages 622–629, 2000.

[10] A. Crétual and F. Chaumette. Image-based visual servoing by integration of dynamic measurements. In *IEEE Conference on Robotics and Automation*, volume 3, pages 1994–2001, Leuven, Belgium, May 1998.

[11] A. Crétual and F. Chaumette. Dynamic stabilization of a pan and tilt camera for submarine image visualization. *Computer Vision and Image Understanding*, 79:47–67, 2000.

[12] R. Cristi, F. A. Papoulias, and A. J. Healey. Adaptive sliding mode control of autonomous underwater vehicle in the dive plane. *IEEE Journal of Oceanic Engineering*, 15(3):152–160, July 1990.

[13] J. P. V. S. da Cunha, R. R. Costa, and L. Hsu. Design of a high performance variable structure position control of ROV's. *IEEE Journal of Oceanic Engineering*, 20(1):42–55, January 1995.

[14] L. De Agapito, R. Hartley, and E. Hayman. Linear calibration of a rotating and zooming camera. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 15–21, Fort Collins, Colorado, June 1999.

[15] L. De Agapito, E. Hayman, and I. L. Reid. Self-calibration of a rotating camera with varying intrinsic parameters. In *British Machine Vision Conference*, pages 751–759, University of Southampton, U. K., September 1998.

[16] K. Deguchi. Optimal motion control for image-based visual servoing by decoupling translation and rotation. In *IEEE/RSJ Intelligent Robots and Systems Conference*, volume 2, pages 705–711, 1998.

[17] R. C. Dorf and R. H. Bishop. *Modern control systems*. Addison-Wesley, 7th edition, 1995.

[18] M. W. Dunnigan and G. T. Russel. Evaluation and reduction of the dynamic coupling between a manipulator and an underwater vehicle. *IEEE Journal of Oceanic Engineering*, 23:260–273, July 1998.

[19] B. Espiau. Effect of camera calibration errors on visual servoing in robotics. In T. Yoshikawa and F. Miyazaki, editors, *Experimental Robotics III*, pages 182–192. Springer-Verlag, 1993.

[20] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–325, 1992.

[21] O. Faugeras. *Three-dimensional computer vision: a geometric viewpoint*. MIT Press, 1993.

[22] O. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(3):485–508, 1988.

[23] T. I. Fossen. *Guidance and Control of Ocean Vehicles*. John Wiley & Sons, 1994.

[24] T. I. Fossen and M. Blanke. Nonlinear output feedback control of underwater vehicle propellers using feedback from estimated axial flow. *IEEE Journal of Oceanic Engineering*, 25(2):241–255, April 2000.

[25] T. I. Fossen and O.-E. Fjellstad. Robust adaptive control of underwater vehicles: a comparative study. In *Proceedings of the Third IFAC Workshop on Control Applications in Marine Systems*, May 1995.

[26] T. I. Fossen and S. I. Sagatun. Adaptive control of nonlinear underwater robotic systems. In *IEEE Conference on Robotics and Automation*, volume 2, pages 1687–1694, April 1991.

[27] A. Fusiello. *Three-dimensional vision for structure and motion estimation*. PhD thesis, Università degli Studi di Trieste, November 1998.

[28] A. Fusiello, E. Trucco, T. Tommasini, and V. Roberto. Improving feature tracking with robust statistics. *Pattern Analysis and Applications*, 2(4):312–320, 1999.

[29] A. J. Fyfe. Planar motion mechanism experiments to determine the stability and control derivatives of the unmanned, cable-controlled submersible, ANGUS. Technical report, Institute of Offshore Engineering, Heriot-Watt University, 1978.

[30] Galil. *DMC-1380 Technical reference guide version 1.2*.

[31] K. R. Goheen and E. R. Jefferys. Multivariable self-tuning autopilots for autonomous and remotely operated underwater vehicles. *IEEE Journal of Oceanic Engineering*, 15(3):144–151, July 1990.

[32] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[33] R. I. Hartley. In defense of the eight-point algorithm. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(6):580–593, June 1997.

[34] A. J. Healey and D. Lienard. Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles. *IEEE Journal of Oceanic Engineering*, 18(3):327–339, July 1993.

[35] A. J. Healey, S. M. Rock, S. Cody, D. Miles, and J. P. Brown. Toward an improved understanding of thruster dynamics for underwater vehicles. *IEEE Journal of Oceanic Engineering*, 20(4):354–361, October 1995.

[36] A. Heyden and K. Aström. Algebraic varieties in multiple view geometry. In *Fourth European Conference on Computer Vision*, volume 2, pages 671–682, 1996.

[37] A. Heyden and K. Aström. Euclidean reconstruction from image sequences with varying and unknown focal length and principal point. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 438–443, 1997.

[38] T. S. Huang and O. D. Faugeras. Some properties of the E matrix in two-view motion estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(12):1310–1312, December 1989.

[39] S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5):651–669, 1996.

[40] I. Idigoras. Interfacing a Cartesian robot to a VME multi-processor real-time computer: Design, implementation and test. Master's thesis, Department of Electrical and Electronic Engineering, Heriot-Watt University, Edinburgh, UK, 1999.

[41] H. K. Khalil. *Nonlinear systems*. Prentice Hall, 2nd edition, 1996.

[42] D. M. Lane. *The investigation of a knowledge based system architecture in the context of a subsea robotic application*. PhD thesis, Department of Electrical and Electronic Engineering, Heriot-Watt University, Edinburgh, UK, 1986.

[43] J.-M. Lavest, G. Rives, and J.-T. Lapreste. Underwater camera calibration. In *European Conference on Computer Vision*, volume 2, pages 654–668, 2000.

[44] K. N. Leabourne, S. M. Rock, S. D. Fleischer, and R. Burton. Station keeping of an ROV using vision technology. In *MTS/IEEE OCEANS*, pages 634–640, 1997.

[45] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(10):133–135, 1981.

[46] J.-F. Lots, D. M. Lane, and E. Trucco. Application of 2 1/2 D visual servoing to underwater vehicle station-keeping. In *MTS/IEEE OCEANS 2000 Conference*, volume 2, pages 1257–1264, September 2000.

[47] J.-F. Lots, D. M. Lane, E. Trucco, and F. Chaumette. A 2-d visual servoing for underwater vehicle station keeping. In *IEEE Conference on Robotics and Automation*, volume 3, pages 2767–2772, 21-26 May 2001.

[48] Q.-T. Luong and O. D. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computer Vision*, 22(3):261–289, 1997.

[49] E. Malis. *Contributions à la modélisation et à la commande en asservissement visuel*. PhD thesis, IRISA, Université de Rennes I, 1998.

[50] E. Malis, F. Chaumette, and S. Boudet. 2 1/2 D visual servoing. *IEEE Trans. on Robotics and Automation*, 15(2):238–250, April 1999.

[51] E. Marchand and G. D. Hager. Dynamic sensor planning in visual servoing. In *IEEE Proceedings of the International Conference on Robotics and Automation*, volume 3, pages 1988–1993, Leuven, Belgium, May 1998.

[52] R. L. Marks, H. H. Wang, M. J. Lee, and S. M. Rock. Automatic visual station keeping of an underwater robot. *IEEE/MTS Oceans Conference*, 2:137–142, 1994.

[53] S. J. Maybank and O. D. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–152, August 1992.

[54] P. Meer, D. Mintz, A. Rosenfeld, and D. Y. Kim. Robust regression methods for computer vision: a review. *International Journal of Computer Vision*, 6(1):59–70, 1991.

[55] Motorola Inc. *MVME166/167/187 single board computers programmer's reference guide*.

[56] Motorola Inc. *PSOS+/68K real-time executive with MMU. User's manual*.

[57] Motorola Inc. *System V/68 release 3. Programmer's guide*.

[58] Motorola Inc. *System V/68 release 3. Programmer's reference manual*.

[59] Motorola Inc. *System V/68 release 3. User's guide*.

[60] Motorola Inc. *System V/68 release 3. User's reference manual*.

[61] S. Negahdaripour, X. Xu, and L. Jin. Direct estimation of motion from sea floor images for automatic station-keeping of submersible platforms. *IEEE Journal of Oceanic Engineering*, 24(3):370–382, July 1999.

[62] J. N. Newman. *Marine Hydrodynamics*. The MIT Press, 1977.

[63] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6(4):348–365, December 1995.

[64] M. Perrier and C. Canudas de Wit. Experimental comparison of PID vs PID plus nonlinear controller for subsea robots. *Autonomous Robots*, 3(2-3):195–212, 1996.

[65] J. Piepmeier, G. McMurray, and H. Lipkin. A dynamic quasi-newton method for uncalibrated visual servoing. In *IEEE Conference on Robotics and Automation*, pages 1595–1600, 1999.

[66] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 90–95, 1998.

[67] M. Pollefeys and L. Van Gool. A stratified approach to metric self-calibration. In *IEEE International Conference on Computer Vision and Pattern Recognition*, pages 407–412, 1997.

[68] M. Pollefeys and L. Van Gool. Stratified self-calibration with the modulus constraint. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(8):707–724, August 1999.

[69] P. Rives and J.-J. Borrelly. Visual servoing techniques applied to an underwater vehicle. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 1851–1856, Albuquerque, NM, USA, April 1997.

[70] L. Robert. Camera calibration without feature extraction. In *IEEE Conference on Computer Vision & Image Processing: Pattern Recognition*, volume 1, pages 704–706, 1994.

[71] P. J. Rousseeuw and A. M. Leroy. *Robust regression and outlier detection.* Wiley, 1987.

[72] C. Samson, B. Espiau, and M. Le Borgne. *Robot Control: the Task Function Approach.* Oxford University Press, 1990.

[73] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, 2000.

[74] L. S. Shapiro, H. Wang, and J. M. Brady. A matching and tracking strategy for independently moving objects. In *Proceedings of the British Machine Vision Conference*, pages 308–315, 1992.

[75] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, June 1994.

[76] Y. Shirai and H. Inoue. Guiding a robot by visual feedback in assembling tasks. *Pattern Recognition*, 5:99–108, 1973.

[77] SIG BERGHER LAHR. *Automation, Assembly facilities, Industrial robots, Positional axes.*

[78] SIG BERGHER LAHR. *System Solutions, positioning systems in wall-mounting technologies.*

[79] C. E. Smith, S. A. Brandt, and N. P. Papanikolopoulos. Eye-in-hand robotic tasks in uncalibrated environments. *IEEE Trans. on Robotics and Automation*, 13:903–914, December 1997.

[80] C. E. Smith and N. P. Papanikolopoulos. Grasping of static and moving objects using a vision-based control approach. *Journal of Intelligent and Robotic Systems*, (19):237–270, 1997.

[81] G. Strang. *Linear algebra and its applications.* Harcourt Brace, 1988.

[82] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical

Report CMU-CS-91-132, Carnegie Mellon University, Pittsburg, PA, April 1991.

[83] E. Trucco, Y. Petillot, I. Tena Ruiz, C. Plakas, and D. M. Lane. Feature tracking in video and sonar subsea sequences with applications. *Computer Vision and Image Understanding*, 79(1):92–122, July 2000.

[84] E. Trucco and A. Verri. *Introductory techniques for 3-D computer vision*. Prentice Hall, 1998.

[85] R. Y. Tsai. Versatile camera calibration technique for high-accuracy 3-D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, RA-3:323–344, August 1987.

[86] R. Y. Tsai, T. S. Huang, and W. L. Zhu. Estimating three-dimensional motion parameters of a rigid planar patch, II: Singular Value Decomposition. *IEEE Trans. On Acoustics, Speech, Signal Processing*, 30:525–534, August 1982.

[87] A. Verri and T. Poggio. Motion field and optical flow: qualitative properties. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 11(5):490–498, May 1989.

[88] L. E. Weiss, A. C. Sanderson, and C. P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE Journal of Robotics and Automation*, RA-3(5):404–417, October 1987.

[89] L. L. Whitcomb and D. R. Yoerger. Development, comparison, and preliminary experimental validation of nonlinear dynamic thruster models. *IEEE Journal of Oceanic Engineering*, 24(4):481–494, October 1999.

[90] L. L. Whitcomb and D. R. Yoerger. Preliminary experiments in model-based thruster control for underwater vehicle positioning. *IEEE Journal of Oceanic Engineering*, 24(4):495–508, October 1999.

[91] W. J. Wilson, C. C. W. Hulls, and G. S. Bell. Relative end-effector control using Cartesian position based visual servoing. *IEEE Trans. on Robotics and Automation*, 12(5):684–696, 1996.

[92] P. Wunsch and G. Hirzinger. Real-time tracking of 3-D objects with dynamic handling of occlusion. In *Proceedings of the International Conference on Robotics and Automation*, volume 4, pages 2868–2873, Albuquerque, N.M., USA, April 1997.

[93] G. Xu and Z. Zhang. *Epipolar geometry in stereo, motion and object recognition*. Kluwer Academics Publishers, 1996.

[94] D. R. Yoerger, J. G. Cooke, and J. E. Slotine. The influence of thrusters dynamics on underwater vehicle behavior and their incorporation into control system design. *IEEE Journal of Oceanic Engineering*, 15(3):167–178, July 1990.

[95] D. R. Yoerger, J. B. Newman, and J. E. Slotine. Supervisory control system for the Jason ROV. *IEEE Journal of Oceanic Engineering*, 11(3):392–399, July 1986.

[96] J. Yuh. Modeling and control of underwater robotic vehicles. *IEEE Trans. on Systems, Man, and Cybernetics*, 20(6):1475–1483, November 1990.

[97] P. Zanne, G. Morel, and F. Piestan. Robust vision based 3D trajectory tracking using sliding mode control. In *IEEE Conference on Robotics and Automation*, volume 3, pages 2088–2093, 2000.

[98] C. Zeller. *Calibration projective affine et euclidienne en vision par ordinateur*. PhD thesis, École Polytechnique, February 1996.

[99] Z. Zhang. Determining the epipolar geometry and its uncertainty: a review. *International Journal of Computer Vision*, 27(2):161–195, 1998.

[100] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *International Conference on Computer Vision (ICCV'99)*, pages 666–673, Corfu, Greece, September 1999.

[101] Z. Zhang and A. R. Hanson. Scaled Euclidian 3-D reconstruction based on externally uncalibrated cameras. In *Proceedings of the IEEE Symposium of Computer Vision*, pages 37–42, November 1995.

[102] S. Ziani-Cherif, G. Lebret, and M. Perrier. Identification and control of a submarine vehicle. In *Proceedings of the Fifth IFAC Symposium on Robot Control, SYROCO'97*, pages 307–311, 1997.