ELSEVIER

Robotics and Autonomous Systems 39 (2002) 145–155

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

Visual station keeping for floating robots in unstructured environments☆

Sjoerd van der Zwaan, Alexandre Bernardino, José Santos-Victor*

Instituto Superior Técnico, Instituto de Sistemas e Robótica, 1049-001 Lisboa, Portugal

Abstract

This paper describes the use of vision for navigation of mobile robots floating in 3D space. The problem addressed is that of automatic station keeping relative to some naturally textured environmental region. Due to the motion disturbances in the environment (currents), these tasks are important to keep the vehicle stabilized relative to an external reference frame. Assuming short range regions in the environment, vision can be used for local navigation, so that no global positioning methods are required. A planar environmental region is selected as a visual landmark and tracked throughout a monocular video sequence. For a camera moving in 3D space, the observed deformations of the tracked image region are according to planar projective transformations and reveal information about the robot relative position and orientation w.r.t. the landmark. This information is then used in a visual feedback loop so as to realize station keeping. Both the tracking system and the control design are discussed. Two robotic platforms are used for experimental validation, namely an indoor aerial blimp and a remote operated underwater vehicle. Results obtained from these experiments are described.

Keywords: Visual tracking; Optic flow; Planar projective motion models; Visual servoing; Underwater robots; Blimp; Station keeping

1. Introduction

Visual control loops have been introduced in order to increase the flexibility and the accuracy of mobile robots. Most cases however deal with the navigation of wheeled mobile platforms that are restricted to the ground plane. More complex systems deal with what we call floating robots that move in 3D space. As an example, we recall the research on the utilization of unmanned aerial vehicles, which has grown with an increasing interest on robotic airships, also known as blimps or lighter-than-air vehicles [1–3]. The motivation behind it is that airships outperform airplanes and helicopters in low-speed, low-altitude applications, having an enormous potential for tasks like environmental and traffic monitoring, climate research, transportation, etc. Yet another example is underwater exploration, where an increase of interest is noted on the utilization of autonomous underwater vehicles (AUVs) and remote operated underwater vehicles (ROVs) [6–8]. These can be inserted into a wide variety of applications related to underwater management, monitoring, inspection and manipulation tasks.

The work presented is integrated in the NARVAL[1] project, for which one of the main goals is the design and implementation of reliable navigation systems for an underwater ROV in unstructured environments. An

☆ This work has been funded by ESPRIT-LTR Project 30185, NARVAL.

* Corresponding author.
E-mail addresses: sjoerd@isr.ist.utl.pt (S. van der Zwaan), alex@isr.ist.utl.pt (A. Bernardino), jasv@isr.ist.utl.pt (J. Santos-Victor).

[1] ESPRIT-LTR Project 30185, NARVAL—Navigation of Autonomous Robots via Active Environmental Perception. www.isr.ist.utl.pt/vislab/NARVAL/index.htm.

0921-8890/02/$ – see front matter
PII: S0921-8890(02)00200-2

indoor blimp was acquired as a testbed for laboratory experiments, simulating the ROV's motion and control degrees of freedom. The problem addressed is that of automatic station keeping based on visual input. The station keeping task is defined locally in the neighborhood of some visually observable landmark and consists in stabilizing the vehicle relative to this landmark while rejecting external disturbances. For floating robots, staying fixed at some given position is not inherent since it is susceptible to significant drift.

A selected image region is used as a visual landmark, whose temporal changes, induced by the vehicle's motion, are tracked. Most environmental scenarios can be reasonably approximated as piecewise planar surfaces. We therefore assume that the camera images a planar surface so that inter-image deformations are completely described by planar projective transformations. The tracker system determines camera motion from the registration between the current live image and an initial reference image. In a prediction phase, optic flow information is used, providing the tracker with an initial estimate of the current image motion parameters. This estimate is then refined using a template matching procedure. This information then provides an input to the station keeping controller. The control objective is to drive the robot back to the desired view under external disturbances, thus assuring some particular alignment of the robot relative to an environmental region. The main difficulties are related to the vehicle's motion constraints, having a limited number of controllable degrees of freedom.

The paper is organized as follows. Section 2 gives some background on multiple view geometry for a pin-hole camera. In Section 3, the tracker system is described and in Section 4 the experimental robotic platforms are presented and modeled. We then turn to the control problem in Section 5 and introduce the visual station keeping controller. In Section 6, the experimental results are described and finally in Section 7 conclusions are drawn and future work is indicated.

2. Background: multiple view geometry

In this section we assume the reader to be familiar with the basic concepts and properties of projective geometry [15,16].

2.1. Camera model

The camera model used in this paper is the standard pin-hole model, which performs a linear projective mapping of the 3D world into the image plane. We also assume that the camera calibration has been performed on beforehand, and that the 3×3 matrix K containing the intrinsic parameters has been estimated. With the pin-hole model, planar image motions cannot be adequately modeled by simple transforms, like affine or translational. A projective planar transformation is the exact motion model when a camera rotates about its eyepoint or if the imaged surface is planar.

2.2. Planar projective transformations

The 2D projective transformation is given by the 3×3 homography, H. This transformation is on image points, and relates points in different views according to $\boldsymbol{x}' = H\boldsymbol{x}$, where $\boldsymbol{x}'$ and $\boldsymbol{x}$ are the homogeneous coordinates of the image points (x', y') and (x, y), respectively. This transformation is defined up to a scale factor and therefore has eight degrees of freedom, given by the entries of H. The computation of a planar transformation requires at least four pairs of corresponding points. In the case of more than four correspondences, a straightforward least-squares linear estimation can be accomplished.

2.3. Image registration

Given a reference image or *template* T and a target image I, the image registration problem is defined as computing a transformation that relates points (x', y') in the template image to points (x, y) in the current target image. Usually, these transformations, are parameterized as a function of a vector $\boldsymbol{q}$, such that $(x', y') = \mathcal{H}_{\boldsymbol{q}}(x, y)$. This transformation is on image coordinates and therefore defines an image warping that maps pixel intensity values from the template image T to the current target image I: $\mathcal{W}(\boldsymbol{q}, T) \mapsto I$. Here, $\mathcal{W}(\boldsymbol{q}, T)$ specifies the image warping according to the transformation parameters $\boldsymbol{q}$.

To register the current image with the template, the best possible match can be obtained through the minimization of an error function, using an appropriate norm, such as the sum-of-squared-differences ($L2$-error criterion). Writing images as column

vectors, the estimate of the current transformation parameters at each time step is then found as:

$$\hat{\boldsymbol{q}} = \arg\min_{\boldsymbol{q}} \left(\tfrac{1}{2} \| I - \mathcal{W}(\boldsymbol{q}, T) \|^2 \right). \quad (1)$$

When iteratively tracking an image region through a video sequence, at each time instant, an initial guess of the current transformation parameters is given by the parameters of the previous step. This provides a first step towards the solution so that only small adjustments remain to be made. In such a scheme, an approximate error criterion is given by:

$$\Delta\hat{\boldsymbol{q}} = \arg\min_{\Delta\boldsymbol{q}} \left(\tfrac{1}{2} \| I_{\boldsymbol{q}_0} - \mathcal{W}(\Delta\boldsymbol{q}, T) \|^2 \right), \quad (2)$$

where $I_{\boldsymbol{q}_0} = \mathcal{W}^{-1}(\boldsymbol{q}_0, I)$ is the image obtained from the inverse warp that maps the current image I approximately onto the template T, according to the initial guess $\boldsymbol{q}_0$. Upon minimizing this criterion, we look for the best residual warp, $\mathcal{W}(\Delta\boldsymbol{q}, T)$ that accounts for the observed difference between the image $\mathcal{W}^{-1}(\boldsymbol{q}_0, I)$ and the template T. The current transformation parameters are then updated according to:

$$\hat{\boldsymbol{q}} = \Delta\hat{\boldsymbol{q}} \otimes \boldsymbol{q}_0,$$

where $\otimes$ stands for the update operator, which is equivalent to homography multiplications.

2.4. Scaled Euclidean reconstruction

Given an inter-image homography, it is possible to reconstruct the relative displacement of the camera in 3D space, up to a scale factor. This is also known as *scaled Euclidean reconstruction* and allows to reconstruct the relative camera trajectory from image registering through a monocular video sequence. This decomposition is described in [15], relating the homography matrix H with the camera rotation, translation and the world plane which induces the homography. The decomposition is the following:

$$H_{21} = K \left(R_{21} + n_1 \frac{t^{\mathrm{T}}}{d_1} \right) K^{-1}, \quad (3)$$

where R_{21} and t are, respectively, the 3×3 rotation matrix and the 3×1 translation vector relating the two 3D camera frames. The world plane is accounted for through the unitary vector n_1, containing the outward plane normal expressed in the camera 1 coordinates, and the distance d_1 of the plane to the first camera center, measured along the optical axis.

3. Tracking system

The tracking system for the station keeping controller aims at tracking a naturally textured landmark in the image plane, whose temporal deformations are then used to recover image and/or camera motion. Upon initialization, an image region is selected as a natural landmark and tracked throughout the video sequence. For each new frame, a prediction of the current incremental transform parameters is obtained from optic flow information. This estimate is then refined by matching the image with a *template* or *reference* image, using a set of pre-defined motion models. The estimated image motion is then iteratively included in the set of motion models so as to sample for future image deformations, likely to occur at the next iteration.

3.1. Prediction phase: optic flow

We include optic flow information in a prediction phase by adjusting an affine model to the observed image motion. The affine motion estimate is computed from the temporal and spatial derivatives in the current and previous live images [12,13]. The advantages are two-fold: (i) by adding information to the initial guess, the residual transformation parameters are kept small; (ii) optic flow provides a means to keep track of the transformation parameters when the visual landmark gets out of the image. Upon integrating the inter-image transform estimates over time, errors are likely to be accumulated. Therefore, keeping track of transformation parameters using optic flow information can be thought of as a means of visual *dead-reckoning* or *odometry*. Accumulated errors are then reset by matching the current image with the template image.

3.2. Update phase: template matching

To register the current image with the template image, we minimize the error function in (2), using a set of m motion models $\{\Delta\boldsymbol{q}_i : i \in (1, \ldots, m)\}$ that sample the parameter space for expected image deformations. Each motion model, $\Delta\boldsymbol{q}_i$, transforms the template image T into an image $\mathcal{W}(\Delta\boldsymbol{q}_i, T)$ that

contains image deformations expected to be observed over time. In our implementation, the algorithm samples into the directions of the individual parameters of the transform parameterization, over varying ranges.

The residual transformation parameters that are looked for, $\Delta \boldsymbol{q}$, can be expressed as a linear combination of the various motion models, $\Delta \boldsymbol{q} = \sum_{i=1}^{m} k_i \Delta \boldsymbol{q}_i$. The image warping operator can now be considered to be specified by the parameter vector $\boldsymbol{k} = [k_1, \ldots, k_m]^{\mathrm{T}}$. The new parameterization is given by:

$$\mathcal{W}(\boldsymbol{k}, T) = \mathcal{W}\left(\sum_{i=1}^{m} k_i \Delta \boldsymbol{q}_i, T\right), \tag{4}$$

where $\mathcal{W}(\boldsymbol{k}, T)$ is the image obtained from warping the template T according to the linear combination of motion models $\Delta \boldsymbol{q}_i$. Substituting (4) into the error function (2), the matching problem can be formulated as finding the linear combination of motion models that best accounts for the observed difference between the approximately registered current image and the template:

$$\boldsymbol{k} = \arg\min_{\boldsymbol{k}} \left(\tfrac{1}{2} \| I_{\boldsymbol{q}_0} - \mathcal{W}(\boldsymbol{k}, T) \|^2\right). \tag{5}$$

The image $\mathcal{W}(\boldsymbol{k}, T)$ is in general a complex and highly nonlinear function of the transformation parameters and the texture map defined in the template image. In order to minimize this error function, we approximate $\mathcal{W}(\boldsymbol{k}, T)$ with a first order Taylor expansion, for small deviations about $\boldsymbol{k} = \boldsymbol{0}$:

$$\mathcal{W}(\boldsymbol{k}, T)|_{\boldsymbol{k}=0} \approx T + \sum_{i=1}^{m} k_i \left.\frac{\partial \mathcal{W}(\boldsymbol{k}, T)}{\partial k_i}\right|_{\boldsymbol{k}=0},$$

where discrete approximations of each partial derivative can be expressed as:

$$\left.\frac{\partial \mathcal{W}(\boldsymbol{k}, T)}{\partial k_i}\right|_{\boldsymbol{k}=0} = \mathcal{W}(\boldsymbol{q}_i, T) - T = B_i.$$

In [4], the set of vectors B_i are denoted *difference templates* and are also used for image registration, but they are justified in a different form. Computing each difference image, B_i, according to the motion model $\boldsymbol{q}_i$, and stacking them into a partial derivatives matrix: $B = [B_1, \ldots, B_m]$, the image $\mathcal{W}(\boldsymbol{k}, T)$ can then be approximated by:

$$\mathcal{W}(\boldsymbol{k}, T)|_{\boldsymbol{k}=0} \approx T + B\boldsymbol{k}.$$

Substituting this approximation into the error function in (5), a least-square solution can be computed for $\boldsymbol{k}$:

$$\boldsymbol{k}_{LS} = (B^{\mathrm{T}} B)^{-1} B^{\mathrm{T}} D, \tag{6}$$

where $D = (\mathcal{W}^{-1}(\boldsymbol{q}_0, I) - T)$ is the observed difference between the approximately registered current image and the template image. After determining $\boldsymbol{k}$, the solution for $\Delta \boldsymbol{q}$ can be calculated.

Most computational requirements go out with the computation of the pseudo-inverse, $(B^{\mathrm{T}} B)^{-1} B^{\mathrm{T}}$, which can be calculated off-line since it is constructed from the set of motion models and the template image. The only on-line computation is the calculation of the difference image, D, implying an image warp $\mathcal{W}^{-1}(\boldsymbol{q}_0, I)$. This makes the method very well-suited for real time tracking applications.

3.3. Adaptive motion models

The choice of the motion models greatly determines the performance of the tracking algorithm. Ideally, this choice should be adapted to the camera motion. This idea has been explored in our implementation of the tracker system, where we adapt the set of motion models according to the history of past detected, inter-image transformation updates. These updates point out into the *direction* and *range* of expected deformations in near future.

An additional small subset is added to the already existing set of motion models and is iteratively adapted to the observed image motion. Within this memory, some heuristic need to be defined to decide which motion model is to be substituted after each iteration. In our case, we replace the least-weighted model, as resulting from the optimization procedure in (6). After initialization, the memory identifies the principal components of image motion. Maintaining the original set intact prevents the algorithm from loosing its ability to sample for deformations in all directions. It follows [9] that upon iteratively substituting motion models, the algorithm is able to track increasing inter-image deformations over a much wider range, thus adding robustness to the system.

When adapting the set of motion models, new difference templates need to be included into the partial derivatives matrix, B, implying on-line calculation of its pseudo-inverse $(B^{\mathrm{T}} B)^{-1} B^{\mathrm{T}}$, thus highly increasing the computational demands. To avoid this, we take

advantage of the information already stored in the pre-calculated pseudo-inverse and update it according to the substituted difference image. It is found [9] that this update can be computed at negligible extra computational effort.

3.4. Tracking performance

With this algorithm, we were able to successfully track a visual landmark undergoing planar projective transformations. A 15 Hz tracking frequency is reached for images with a 128×192 pixel size, using an off-the-shelf 450 MHz processor. Fig. 1 shows results of tracking an image region in submarine images. The initially selected image region is used as a template, whose temporal deformations are tracked over time.

3.5. Optimal landmark selection

When selecting an image region as a template for tracking, its texture map should contain sufficient information so that expected image deformations over time can be observed from it. To automatically select a template from an image, some optimality criterion needs to be evaluated, that takes the observability with respect to the motion models into account. To do so, we follow the approach in [5], and model the observed difference, $D = \mathcal{W}^{-1}(\boldsymbol{q}_0, I) - T$, as a linear combination of the pre-calculated difference images, in the presence of additive noise:

$$D = B\boldsymbol{k} + u, \tag{7}$$

where u is additive noise, $\boldsymbol{k}$ represents the real transformation parameter vector that is looked for and B is the partial derivatives matrix containing all difference images. The least-square estimate for $\boldsymbol{k}$ is given in (6) and can be rewritten using (7) as:

$$\boldsymbol{k}_{LS} = \boldsymbol{k} + ((B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}})u. \tag{8}$$

In order to have $\boldsymbol{k}_{LS}$ as a reliable estimate of $\boldsymbol{k}$, we would like to choose a B, such that the uncertainty introduced by $((B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}})u$ is minimized. The partial derivative matrix B is a function of the selected template texture and the set of motion models. For the same set of motion models, different templates result in different values of uncertainty.

To measure this uncertainty, we take the $L2$-norm on the error in the reconstructed signal:

$$\|\boldsymbol{k} - \boldsymbol{k}_{LS}\|^2 = \|((B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}})u\|^2. \tag{9}$$

Assuming zero-mean, unit variance white noise for u, we can take the expected value of (9), which can be computed as:

$$E\{\|((B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}})u\|^2\} = \mathit{trace}((B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}}). \tag{10}$$

The optimal template is then found by minimizing the expected value of (10), given the set of motion models.

Fig. 2 shows the most and less informative template found in an underwater image, for a fixed size landmark. The selection of informative landmarks has a noticeable impact on the tracking accuracy. Some test

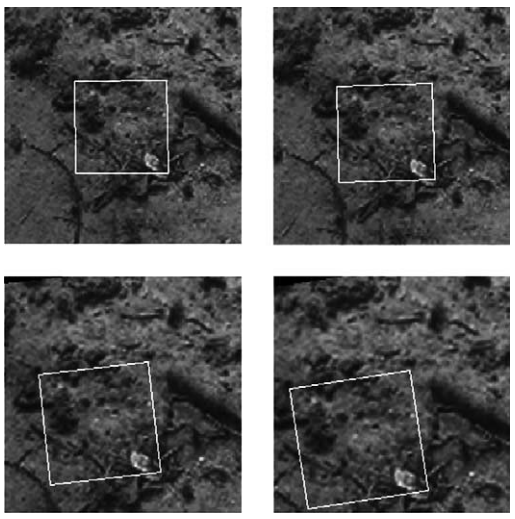

Fig. 1. Tracking an image region in a submarine video sequence, under planar projective image deformations.

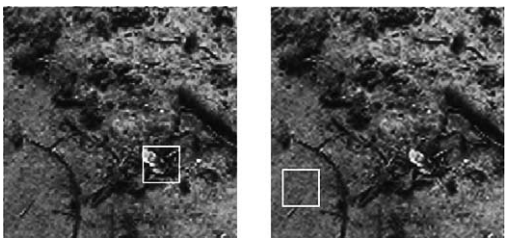

Fig. 2. Automatic landmark selection: (left) most informative image region; (right) less informative image region.

were performed to evaluate the tracking errors, resulting in sub-pixel accuracy when tracking informative image regions.

4. Robot description and modeling

Before turning to the control design, we first describe the experimental robotic platforms used, namely an underwater ROV and an indoor blimp.

4.1. Floating robots: ROV and blimp

A commercially available Phantom 500SP ROV is used, which is adapted for computer control. The ROV is illustrated in Fig. 3 and is equipped, among other sensors, with an on-board pan and tilt camera. The camera is mounted rigidly to the ROV, such that its optical axis is aligned with the vertical axis of the ROV reference frame. The pan and tilt angles can be controlled separately, resulting in two extra degrees of freedom for the camera. The ROV is wired to a remote processing unit by a 150 m umbilical. Video signals are sent up to the ground surface. Here, control signals are derived and sent down to the ROV via the umbilical, through a serial communication link.

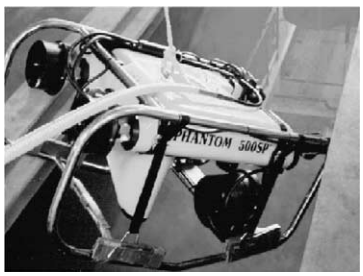

Fig. 3. Computer controlled Phantom ROV with an on-board pan and tilt camera.

The small-size indoor blimp is illustrated in Fig. 4. It is composed of a balloon, a gondola and a remote controller. For the blimp to float in air, its envelope needs to be filled with a gas that is lighter then air, typically helium, providing it with sufficient payload to carry the gondola, batteries and camera. The gondola is a rigid structure attached to the bottom of the balloon. It contains the motor controllers, a radio receiver and the three thrusters for propulsion in the horizontal and vertical planes. Additionally, a mini camera with a video link was mounted on the gondola. The CCD camera sends video signals to a remote computer via a video link in open air. The images received by the computer are processed and analyzed so as to derive proper control signals, sent to the blimp via a radio link.

In both cases, the controllable degrees of freedom are defined by the geometric arrangement of the thrusters. The vehicles where originally designed for joystick-type piloting, where a forward/backward force and a differential torque are commanded by two horizontally placed thrusters and an upward/downward force is commanded through a vertically placed thruster. With this arrangement, non-holonomic motion constraints are specified, requiring complex maneuvers for posture stabilization.

4.2. Dynamics and kinematics

The dynamic model can be obtained by writing down the six degrees of freedom *Newton–Euler* equations of motion resolved into a body-fixed reference

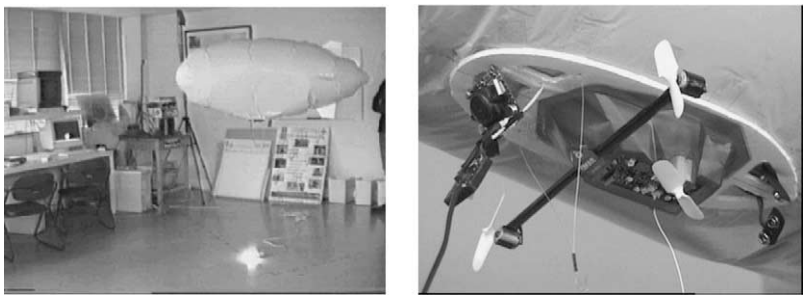

Fig. 4. Radio controlled indoor blimp (left) with on-board camera and video link (right).

frame [2,14]:

$$M\dot{\boldsymbol{v}} + \boldsymbol{c}(\boldsymbol{v}) + D(\boldsymbol{v})\boldsymbol{v} + \boldsymbol{g}(R) = \boldsymbol{\tau}.$$

Here, $\boldsymbol{v}$ is the instantaneous velocity vector containing the linear and angular components of the vehicle velocity, M the mass matrix containing all the masses and inertias of the rigid body, $\boldsymbol{c}(\boldsymbol{v})$ groups the coriolis and centrifugal terms, $D(\boldsymbol{v})$ contains the damping and friction forces, $\boldsymbol{g}(R)$ is the restoring forces vector containing gravitational and buoyancy terms (depending upon the vehicle orientation w.r.t. some inertial world frame) and $\boldsymbol{\tau}$ is the vector of propulsion forces and torques.

The generated forces and torques in $\boldsymbol{\tau}$ are related to the control commands sent to the thrusters by the affine thruster model [14]:

$$\boldsymbol{\tau} = B\boldsymbol{u},$$

where $\boldsymbol{u}$ is the control input vector and B captures the relationship between control commands (typically given as pwm-signals) and generated forces and torques from the thrusters.

Upon integration of the resolved acceleration, the robot instantaneous velocity is obtained, which can be related to the world referenced velocity, leading to the kinematic equations [14]:

$$\dot{\boldsymbol{n}} = J\boldsymbol{v}.$$

Here $\boldsymbol{n}$ contains the position and orientation of the robot in some fixed world frame and J is a Jacobian relating the robot instantaneous velocity to world referenced velocity. The kinematic model is useful for simulation and navigation purposes.

5. Visual station keeping

For station keeping, we assume that the robot is hovering parallel to a piecewise planar ground-plane in the environment, having the camera looking approximately perpendicular to the plane. Motivated by the limited controllable degrees of freedom, a *decoupled control design* is adopted, which station keeps the robot in the horizontal plane w.r.t. the landmark, while maintained at a fixed altitude in the vertical plane. The controller design is addressed to within the framework of visual servoing strategies. A tutorial on this topic can be found in [11]. For navigation purposes, these strategies can be roughly classified into two architectures: (i) position based (or 3D) visual servoing; (ii) image based servoing. In the case of 3D servoing, images are used to reconstruct the scene and estimate 3D positions/orientations from visual information. In the image based approach, features are measured directly from the image plane and used to synthesize the control laws without an intermediate reconstruction phase. Both architectures are discussed and an image based station keeping controller will be proposed and experimentally validated.

5.1. 3D-servoing

When tracking a planar landmark in the image plane, the estimated inter-image homographies can be decomposed into relative camera displacements, as described in Section 2.4. This provides a means of reconstructing the relative camera trajectory in 3D-space over time and self-localize the camera w.r.t. the initial view. To realize station keeping, a kinematic error function can be defined between the current estimated and initial camera pose. The station keeping controller then aims at regulating this error to zero using feedback. Since these errors are defined in Cartesian space, it is relatively easy to obtain a controller design based upon geometric insight.

The main advantage of the 3D approach is that it directly controls the camera trajectory in Cartesian space. However, since the control design is based on error functions in Cartesian space, the tracked landmark used for the reconstruction phase may leave the image and lead to servoing failure. Another drawback is the sensitivity of the homography decomposition w.r.t. tracking errors. Fig. 5 shows results on camera trajectory reconstruction from estimated homographies face to ground-truth. It follows that although tracking errors are kept small in the image plane (less than 1 pixel) and intrinsic parameters are assumed to be known, significant errors in the camera trajectory reconstruction occur. Major errors arise in the reconstruction of relative displacement in the x- and y-direction. This is due to the fact that, under weak perspective distortion, translation in the image plane can be accounted for by either a camera translation or a rotation around the *pan* and *tilt* angles in 3D space, leading to an ambiguity when reconstructing 3D motion from 2D motion.

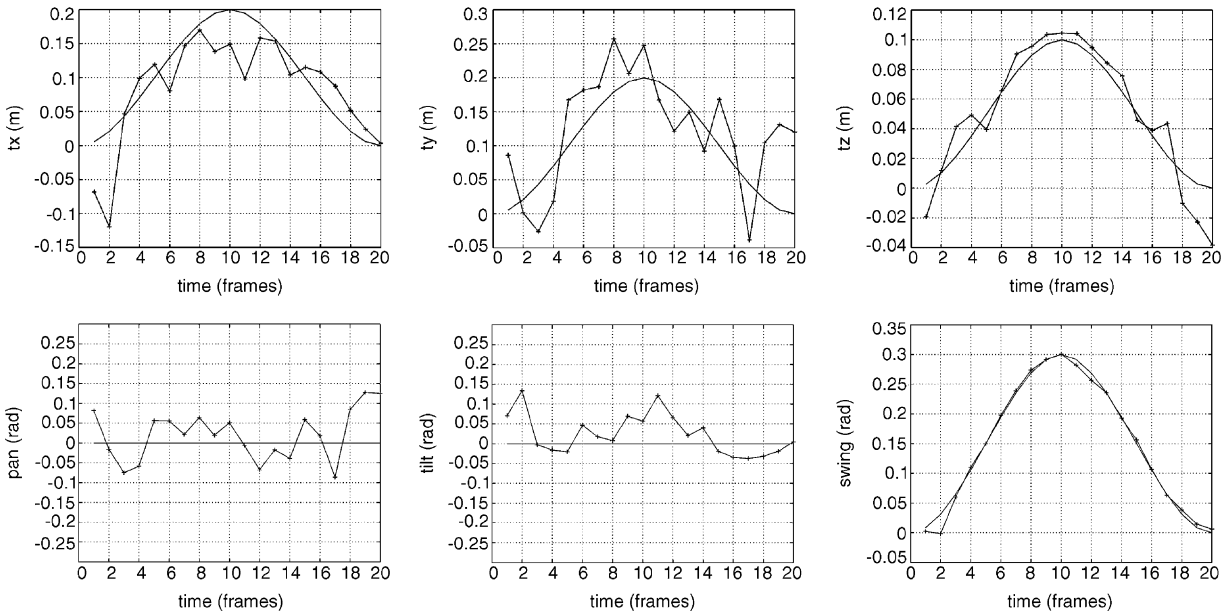

Fig. 5. Results on camera trajectory reconstruction from homography decomposition. Reconstructed signals are +-marked, whereas ground-thruth values are plotted as continuous lines.

5.2. Image based servoing: station keeping in the horizontal plane

The image based station keeping task is defined as the regulation to zero of an image error function $\boldsymbol{e}(\boldsymbol{s}) = \boldsymbol{s} - \boldsymbol{s}_{\mathrm{d}}$, where $\boldsymbol{s}$ is the image feature parameter vector and $\boldsymbol{s}_{\mathrm{d}}$ the desired value. The centroid of a tracked image region is used as a feature, whose desired position is at the image center. The image error function is then given by $\boldsymbol{e} = [x_c, y_c]^{\mathrm{T}} - [x_d, y_d]^{\mathrm{T}}$ and the controller aims at driving the centroid towards the image center.

Changes in the image features can be related to changes in the relative camera pose. This kinematic relationship is often referred to as the *image Jacobian* or the *interaction matrix* [10,11]:

$$\dot{\boldsymbol{s}} = L\boldsymbol{v}_{\mathrm{cam}}, \tag{11}$$

where L is the image Jacobian and $\boldsymbol{v}_{\mathrm{cam}}$ is the 6×1 camera velocity screw. The Jacobian for image points is given by the motion field [11], depending both on the point coordinates and their depth, Z. An exponential decrease of the error function is obtained by imposing $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$, with λ some positive constant. Using (11), we can then solve for the camera motion that guarantees this convergence:

$$\boldsymbol{v}^*_{\mathrm{cam}} = -\lambda L(\boldsymbol{s}, Z)^{+}(\boldsymbol{s} - \boldsymbol{s}_{\mathrm{d}}), \tag{12}$$

where $\boldsymbol{v}^*_{\mathrm{cam}}$ is the resolved camera velocity that drives the centroid to the image center and L^+ is the pseudo-inverse of the image Jacobian.

The robot control inputs are in general defined in the vehicle reference frame, commanding components of the vehicle velocity vector. We therefore need to relate the controllable components of the vehicle velocities to camera velocities. This relationship is given by the control input Jacobian:

$$\boldsymbol{v}_{\mathrm{cam}} = J_{\mathrm{robot}}\bar{\boldsymbol{v}}_{\mathrm{robot}}, \tag{13}$$

where $\bar{\boldsymbol{v}}_{\mathrm{robot}}$ contains the controllable velocity components of the vehicle velocity screw and J_{robot} is the control input Jacobian. This Jacobian depends on the camera position and orientation in the vehicle reference frame and can be easily computed from transforming linear and angular velocity components between the frames. For station keeping, we consider the linear and angular velocity of the vehicle in the horizontal plane, $\bar{\boldsymbol{v}}_{\mathrm{robot}} = [v, \omega]^{\mathrm{T}}$, which are both

controllable from the two back thrusters. Substituting (13) into (11), an expression is obtained that relates image point velocities to the vehicle velocity:

$$\dot{\boldsymbol{s}} = L\, J_{\text{robot}} \bar{\boldsymbol{v}}_{\text{robot}}. \tag{14}$$

With this expression, we can solve for the desired robot velocity in the horizontal plane, necessary to guarantee the convergence of the image error function:

$$\bar{\boldsymbol{v}}^{*}_{\text{robot}} = -\lambda (L(\boldsymbol{s}, Z) J_{\text{robot}})^{+} \boldsymbol{e}. \tag{15}$$

This expression takes the vehicle motion constraints into account, resulting into trajectories that are physically executable.

5.3. Image based servoing: auto-altitude controller

The image based controller for the vertical plane aims at maintaining the robot at a fixed depth during station keeping maneuvers. The controller design is such that it maintains the appearance of the landmark in the image plane at the same scale. Having the robot hovering parallel to a planar region, the scale in the image plane of some selected landmark has a direct physical interpretation in terms of relative depth.

To recover the scale in the image plane, we consider the inter-image homography as a hierarchical chain of transformations on the image plane, as described in [16]. It follows that the scale factor can then be recovered from the determinant of the upper-left non-singular 2×2 block of the estimated homography, for simplicity indicated as A:

$$s = \sqrt{|A|}. \tag{16}$$

Taking this scale as the control error function, the desired vertical control is given by:

$$\bar{\boldsymbol{v}}^{*}_{\text{robot}} = K_p s + K_d \dot{s} + K_i \int s \, \mathrm{d}t, \tag{17}$$

where in this case $\bar{\boldsymbol{v}}_{\text{robot}}$ contains the resolved desired vertical speed and a PID design was adopted for dynamic compensation.

6. Experimental results

To validate the proposed control strategies, a set of real experiments were done initially using the blimp in a laboratory environment. Fig. 6 shows the temporal evolution of the error signals during a docking and station keeping experiment. At the left side, the image trajectory of the target point (centroid of the tracked window) is illustrated under closed-loop control. The control strategy aims at driving this point to the image center (docking) and keep it as close as possible to this center (station keeping). The right image shows

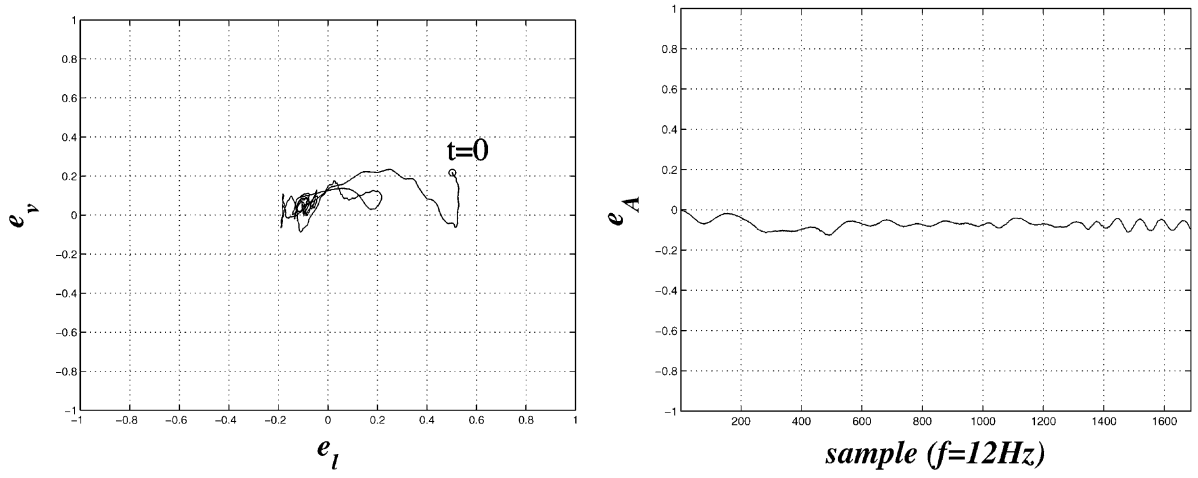

Fig. 6. Station keeping test with the blimp: (left) trajectory of the centroid of the tracked window (image errors in normalized pixel coordinates); (right) difference between the area of the reference window and the tracked window.

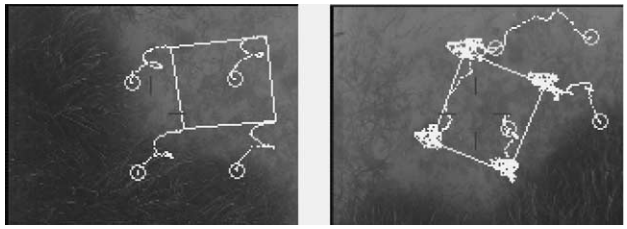

Fig. 7. Station keeping experiment with the ROV at the Mediterranean: (left) tracking a selected image region in the presence of drift, with the ROV uncontrolled; (right) controlling the centroid back to the image center by servoing the vehicle.

the error between the areas of the reference window and the tracked window and indicates that the blimp is approximately maintained at a constant height.

In a later stage, several successful station keeping trials were performed with the ROV at open sea. The system was tested under various environmental conditions at different locations, namely in the North Sea near Orkney, Scotland, as well as in the Mediterranean sea in Villefranche, France. The results of a station keeping test in the Mediterranean sea are shown in Fig. 7. In a first stage, the vehicle floats uncontrolled when a landmark is selected around the image center and tracked in the presence of drift. Note that even with poor texture, the tracker was able to accurately track the selected image region. Then the visual feedback loop is closed and the landmark is driven back towards the image center, where it remains oscillating around the desired position under external disturbances. The evolution of the error signals are shown in Fig. 8 and illustrate the convergence of the errors for the station keeping controller and the auto-depth controller.

For both the blimp and the ROV, no efforts are made so as to control the landmarks orientation towards a desired value. The main difficulties arise for lateral

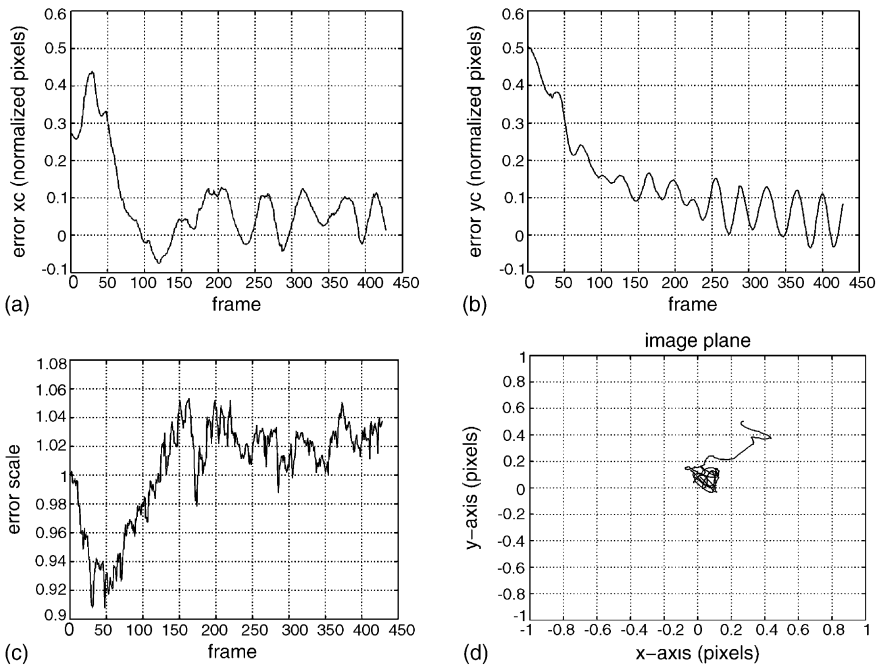

Fig. 8. Evolution of the error signals during the station keeping test with the ROV at open sea: (a) x-coordinate of the centroid; (b) y-coordinate of the centroid; (c) relative scale; (d) centroid trajectory in the image plane.

offsets of the centroid in the image plane. In this case, since the robots have no lateral controllable degrees of freedom, the only solution is to compensate these errors by rotating, resulting into complex curved trajectories of the centroid and the landmark corners in the image plane.

7. Conclusions

In this paper we presented the tracking and control aspects for automatic visual station keeping with floating robots. Tracking of image regions was realized by integrating optic flow information with a template matching method, resulting in subpixels tracking accuracy. Planar projective motion models were considered that cover the whole range of image deformations that occur when a camera moves in 3D. For template matching, a set of motion models was used, sampling for expected image deformations. The main advantage is that these can be pre-calculated when applied to the template image, resulting in high tracking frequencies. To enhance robustness, the set of models was iteratively adapted to the history of detected camera motion. Also a method for automatic landmark selection was described, selecting the most informative image region for tracking.

Using the tracker information, visual control loops were designed to perform station keeping. We encountered serious difficulties in the reconstruction phase of 3D servoing architectures when compared to the more robust image based servoing schemes. The station keeping task was therefore formulated in the image plane and a decoupled control strategy was adopted. For station keeping, we considered the regulation of the landmark centroid towards the image center, while not controlling its orientation towards a final value at all. The main motivation was that, given the vehicle motion constraints, lateral offset in the image plane can only be compensated by rotating the robots.

Although a dynamic model for the robots was derived, the proposed control laws are based on kinematic error functions only. However, deriving the dynamic model gave us a better insight into the system's behavior and the coupling effects between kinematic variables could be identified. For future work, we consider to include the vehicle dynamics into the tracking system and the controller design.

References

[1] A. Elfes, S. Siqueira Bueno, M. Bergerman, J.J.G. Ramos, A semi-autonomous robotic airship for environmental monitoring missions, in: Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 1998.
[2] S.B. Varella Gomes, J.J.G. Ramos, Airship dynamic modeling for autonomous operation, in: Proceedings of the IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 1998.
[3] H. Zhang, J.P. Ostrowski, Visual servoing with dynamics: Control of an unmanned blimp, in: Proceedings of the IEEE International Conference on Robotics and Automation, Detroit, MI, May 1999.
[4] M. Gleicher, Projective registration with difference decomposition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, June 1997, pp. 331–337.
[5] S.J. Reeves, L.P. Heck, Selection of observations in signal reconstruction, IEEE Trans. Signal Process 43 (1995) 788–791.
[6] J.F. Lots, D.M. Lane, E. Trucco, Application of $2\frac{1}{2}$D visual servoing to underwater vehicle station keeping, in: Proceedings of the IEEE Oceans Conference, Providence, RI, September 2000.
[7] P. Rives, J. Borrelly, Visual servoing techniques applied to an underwater vehicle, in: Proceedings of the IEEE International Conference on Robotics and Automation, Albuquerque, NM, April 1997.
[8] R. Garcia, J. Battle, X. Cufi, J. Amat, Positioning an underwater vehicle through image mosaicking, in: Proceedings of the IEEE International Conference on Robotics and Automation, Seoul, Korea, May 2001.
[9] S. van der Zwaan, Vision based station keeping and docking for floating robots, MSc Thesis, Lisbon, May 2001. http://www.isr.ist.utl.pt/labs/vislab/thesis/.
[10] B. Espiau, F. Chaumette, P. Rives, A new approach to visual servoing in robotics, IEEE Transactions on Robotics and Automation 8 (3) (1992) 313–326.
[11] S. Hutchinson, G.D. Hager, P.I. Corke, A tutorial on visual servo control, IEEE Transactions on Robotics and Automation 12 (5) (1996).
[12] J. Santos-Victor, G. Sandini, Visual behaviours for docking, Computer Vision and Image Understanding 67 (3) (1997) 223–238.
[13] M. Subbarao, A. Waxman, Closed form solutions to image flow equations for planar surfaces in motion, Computer Vision Graphics and Image Processing 36 (1986) 208–228.
[14] T.I. Fossen, Guidance and Control of Ocean Vehicles, Wiley, New York, 1995.
[15] O. Faugeras, Three-Dimensional Computer Vision, MIT Press, Cambridge, MA, 1993.
[16] R. Hartley, A. Zisserman, Multiple View Geometry in Computer Vision, Cambridge University Press, Cambridge, 2000.