Divergent Stereo in Autonomous Navigation : From Bees to Robots *

J. Santos-Victor(1), G. Sandini(2), F. Curotto(2), S. Garibaldi(2)

(1) CAPS/ Instituto Superior Técnico
Instituto de Sistemas e Robótica
Complexo 1, Av. Rovisco Pais, 1096 Lisboa codex, Portugal
e-mail: jasv@kappa.ist.utl.pt

(2) DIST - University of Genova
Laboratory for Integrated Advanced Robotics (LIRA - Lab)
Via Opera Pia 11A - I16145 Genoa, Italy
e-mail:giulio@vision.dist.unige.it

Abstract

This paper presents some experiments of a real-time navigation system driven by two cameras pointing laterally to the navigation direction (*Divergent Stereo*). Similarly to what has been proposed in [1, 2], our approach [3, 4] assumes that, for navigation purposes, the driving information is not distance (as it is obtainable by a stereo setup) but motion and, more precisely, by the use of qualitative optical flow information computed over non-overlapping areas of the visual field of two cameras.

Following this idea, a mobile vehicle has been equipped with a pair of cameras looking laterally (much like honeybees) and a controller based on fast, real-time computation of optical flow has been implemented. The control of the mobile robot (*Robee*) is based on the comparison between the apparent image velocity of the left and the right cameras. The solution adopted is derived from recent studies [5] describing the behavior of freely flying honeybees and the mechanisms they use to perceive range.

This qualitative information (no explicit measure of depth is performed) is used in many experiments to show the robustness of the approach, and a detailed description of the control structure is presented to demonstrate the feasibility of the approach in driving the mobile robot within a cluttered environment.

A discussion about the potentialities of the approach and the implications in terms of sensor structure is also presented.

*The research described in this paper has been supported by the Special Projects on Robotics of the Italian National Council of Research and by the ESPRIT project VAP-II. A fellowship from a bilateral collaboration between Consiglio Nazionale delle Ricerche (CNR) and Junta Nacional de Investigação Científica e Tecnológica (JNICT) is gratefully acknowledged by José Santos-Victor.

1 Introduction

Research in computer vision has often been motivated by similarities with solutions adopted in natural living systems. Since the early work in image processing and computer vision, the structure of the human visual system has often been used, as the "natural" model of artificial visual systems. Comparatively less effort has been devoted to the study and implementation of artificial vision systems based on the so-called "lower level" animals. The motivation of this biased research activity stems mainly from the idea that by "understanding human vision" one could obtain a "general" solution to the visual process, and eventually be able to develop a "general purpose visual system".[1] We certainly agree with the idea that there is a lot to learn from human vision (and conversely there is a lot to learn by trying to build artificial systems with anthropomorphic features) but, on the other hand we do not believe that such a "general purpose vision system" actually exists, and we are convinced that a lot can be gained and understood by looking at much simpler biological systems.

Looking more carefully, in fact, it is evident that the human visual system is not as general purpose as it may look. For example, it is of little use underwater, it has a limited spectral band, it cannot measure distance, size or velocity in metric terms. Even its recognition system can be fooled very easily, for example by turning upside-down pictures of even familiar faces. The apparent generality of the system comes from the fact that we actually perform a limited number of motor and cognitive "actions" and, within this limited domain, our visual system (or more generally the integration of our perceptual systems) performs very efficiently.

Following these ideas, one could say that the goal of a vision system in a "living" agent is not generality but specificity: the physical structure and purpose drive perception [6, 7, 8].

Within the scope of this paper we would like to argue that the frontal position of the eyes (with a very large binocular field) is mainly motivated by manipulation requirements and, if one restricts the purpose to navigation control, a potentially more effective positioning is the one adopted by flies, bees and other flying insects, namely with the eyes pointing laterally. The tradeoff between these two extreme situations is that in the latter case the global visual field (i.e. the union of the visual fields of the two eyes) can be enlarged *without increasing the computational power* [2]. On the other hand, by increasing the binocular part of the visual field, the region where a stereo-based depth measure is possible, becomes larger.

Looking again at biology, one finds out that the position of the eyes in different species becomes more frontal as the manipulative abilities increase (it is not a case that humans and primates have, among all species, the wider binocular field and the more frontal positioning).

Other aspects worth considering are the fact that stereo acuity is maximal at short dis-

[1] And, possibly, the notion that "human vision is complex while insect vision is simple".

[2] Of course one could use "technological tricks" like rear-viewing mirrors, but this is not a valid argument in this context.

tances, which, behaviorally correspond to the manipulative workspace. Moreover, stereo is the only visual modality providing depth information in static environments (which is behaviorally relevant particularly in manipulative tasks). Conversely, motion parallax is useful if the "actor" or the environment is dynamic and its accuracy (and the corresponding range of operation) can be tuned by an active observer, by varying its own velocity. In this respect motion-derived features such as time-to-crash, seem more relevant to dynamically control posture and other navigation parameters: if one has to jump over an obstacle, the change in posture while he/she is approaching the obstacle (for example when to start to lift the leg), may be driven by "time-to-crash" more than by distance (which would be dependent upon the speed of approaching).

The biological model of the navigating actor proposed in this paper comes from insects and the use they make of flow information to solve apparently complex motor tasks like flying in unconstrained environments and landing on surfaces[9, 10, 11, 1]. Particularly relevant to this paper is the experiment reported in [5] where honeybees were trained to navigate along corridors in order to reach a source of food. The behavioral observation is the fact that, even if the corridor was wide enought to allow for "irregular" trajectories, bees were actually flying in the middle of the corridor. This finding is even more surprising if we take into consideration that insects do not have accommodation and, that the stereo baseline is so small that disparity cannot be reliably measured. Apparently, then, no depth information can be derived using "traditional" methods. The solution presented in [5], is rather simple and it is based on the computation of the difference between the velocity information computed from the left and the right eyes: if the bee is in the middle of the corridor the two velocities are the same, if the bee is closer to one wall, the velocity of the ipsilateral eye is larger. A simple control mechanism (the so-called *centering effect*), then, may be based on motor actions that tend to minimize this difference: move to the left if the velocity measured by the right eye is larger than that measured by the left (and vice versa).

Following this line of thought, a mobile robot (*Robee*) has been equipped with laterally looking cameras and a controller has been implemented, based upon motion-derived measures, which does not rely on precise geometric information but takes full advantage of the continuous (in time) nature of visual information. Navigation is controlled on the basis of the optical flow field computed over windows of images acquired by a pair of cameras pointing (in analogy with the position of the eyes in the honeybees and other insects) laterally and with non overlapping visual field. We called this camera placement *Divergent Stereo.*

While working on the experiments presented here, we became aware of a similar implementation (the *beebot* system) proposed by Coombs and Roberts [2]. A discussion of the differences between the two approaches will be presented later on. However, the main difference lies on the fact that, while in the system proposed by Coombs and Roberts gaze control is part of the navigation strategy, in the present implementation a simpler setup has been analyzed where gaze control becomes unnecessary by appropriately positioning the two cameras and by controlling some behavioral variables such as the "turning speed".

The experiments reported describe the behavior of *robee* in tasks like: following a wall, avoiding obstacles, making sharp and smooth turns, navigating along a funneled corridor with a very narrow exit.

In Section 2, the *Divergent Stereo* approach proposed for robot navigation will be described along with the experimental setup. The simplified computation of the optical flow is presented in Section 3. The control aspects will be presented in Section 4. Finally, after the presentation of the experimental results (in Section 5), a brief discussion will summarize the major points of the paper within the framework of qualitative vision.

2 The Divergent Stereo Approach

The basis of the visually guided behavior of *robee* is the *centering reflex*, described in [5] to explain the behavior of honeybees flying within two parallel "walls". The qualitative visual measure used is the difference between the image velocities computed over a lateral portion of the left and the right visual fields.

Even if the principle of operation is very simple, a few points are worth discussing, before entering into the implementation details, in order to explain the underlying difficulties and the design principles adopted. The first, and possibly major, driving hypothesis is the use of qualitative depth measurements: no attempt is made to actually compute depth in metric terms. The second guideline is simplicity: whenever possible, the tradeoff between accuracy and simplicity has been biased toward the latter criterion. Finally, the goal of our visuo-motor controller is limited to the "reflexive" level of a navigation architecture acting at short-range. In spite of these limitations, we will demonstrate a variety of navigation capabilities which are not restricted to obstacle avoidance or to the "centering reflex".

Among the practical problems encountered in the actual implementation, two are worth discussing to better appreciate some of the concepts presented later on. The first point regards the relationship between heading direction and optical flow computation. Strictly speaking, flow information can be reliably used for the centering reflex only when the directions of the two cameras are symmetric with respect to the heading direction. With the camera placement of *robee* (see Figure 1) this requirement is satisfied only during translational motion of the robot. During rotational motions (as during obstacle avoidance), the flow field is not solely dependent on the scene structure. This problem has been solved by Coombs and Roberts [2] by stabilizing the cameras against robot rotations and by introducing a control loop keeping the gaze aligned with the heading direction. The solution adopted in *robee* is different, as the two cameras are fixed. Section 3.1 presents a detailed analysis of the rotational field and discusses ways to overcome this problem, either by using special control strategies, or by carefully selecting the camera positions.

A second, relevant, point is the unilateral or bilateral absence of flow information, caused by absence of texture, or by localized changes in environmental structure (e.g. an open door along a corridor)[3]. By applying the centering reflex in a crude mode, these lack of flow information would produce a rather unsteady behavior of the robot. This problem has been solved by introducing a *sustaining* mechanism to stabilize, in time, the unilateral flow amplitude in case of lack of information. This simple qualitative mechanism does not alter the reflexive behavior of *robee* (in the sense that it is neither based on prior knowledge of the environment, nor on metric information) and extends the performance of the system to a much wider range of environmental situations (see Section 4.3 for more details).

The experimental setup is based on a computer controlled mobile platform, *TRC Labmate*, with two cameras pointing laterally. Two cameras mounted with 4.8mm, auto-iris lenses are connected to a VDS 7001 Eidobrain image processing workstation. The left and right images are acquired simultaneously, during the vehicle motion. The setup is illustrated in Figure 1. During the experiments, the vehicle forward speed is approximately 80 mm/s.

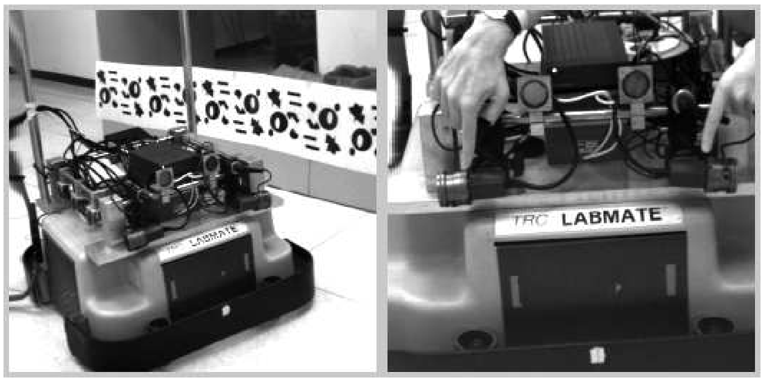

Figure 1: *Robee* with the divergent stereo camera setup

3 Optical Flow Computation

To compare the image velocity seen by the left and right cameras, the average of the optical flow on each side is computed. The optical flow $V = (u, v)$, is usually obtained by using the fundamental optical flow constraint [12], shown in equation (1), or by imposing second order

[3]This situation occurs, even with textured environments, if the relationship between "wall(s)" distance and vehicle speed is such that the resolution of the optical flow computation is lower than image velocity (e.g. if the robot is moving slowly and the "walls" are very far away).

stationarity constraints in the brightness function [13, 14, 15], according to equation (2):

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0 \tag{1}$$

$$\frac{d}{dt}\nabla I = 0\,. \tag{2}$$

By using expression (2), one ends up with the following system of equations, which can be solved to obtain both components of the optical flow.

$$I_{xx}u + I_{xy}v = -I_{xt} \tag{3}$$

$$I_{xy}u + I_{yy}v = -I_{yt} \tag{4}$$

where I_{xx}, I_{yy} and I_{xy} stand for the second spatial derivatives of the image, while I_{xt} and I_{yt} stand for the time-space derivatives.

In order to obtain useful flow estimates, it is necessary, as in other similar approaches, to smooth the images, in both space and time domains. Usually, this is accomplished through the use of gaussian smoothing (in space and time). The temporal smoothing is generally computed, centering the filter kernel in the image to be processed and, as a result, this procedure requires not only past images, but has also to use images to be acquired *after* the time instant under consideration (a non causal filter).

The time delay introduced with this procedure, becomes relevant when the visual information is to be used for real-time control because it introduces a lag in the feedback control loop. In our approach, instead, having the control application in mind, a *causal* first order time filter has been used, which recursively updates the time smoothed image:

$$I_s(0) = I(0) \tag{5}$$

$$I_s(t) = \lambda I_s(t-1) + (1-\lambda)I(t) \qquad 0 \leq \lambda \leq 1 \tag{6}$$

where $I(t)$ and $I_s(t)$ stand for the image acquired at time t and the corresponding *temporal smoothed* image. The parameter λ controls the degree of smoothing desired. When λ is larger, the images are more smoothed in time. Using the recursive time filtering procedure, only present and past images are required for the time filtering process [4].

The experiments used to demonstrate the *Divergent Stereo* navigation concept are based upon a mobile platform moving on a flat floor. As the robot motion is constrained to the ground plane, it can be assumed that the flow along the vertical direction is negligible (unless there is a significant lateral motion, which is seldom the case). Hence, we can use a simpler computation procedure, which is faster and more robust (since, for example, it does not involve

[4] Regarding memory storage, it is only necessary to store $I(t)$, $I_s(t)$ and $I_s(t-1)$.

the computation of second derivatives), by simply assuming in equation (1) that the vertical flow component, v, is 0 :

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0 \qquad (7)$$

$$v = 0 \qquad (8)$$

and u is given by:

$$u = -\frac{I_t}{I_x} \qquad (9)$$

where I_x and I_t stand for the x spatial derivative and the temporal derivative of the image.

The spatial smoothing is performed by convolving the time smoothed images, with a gaussian filter, and the time derivative is simply computed by subtracting two consecutive space and time smoothed images.

To speed up the optical flow computation, a set of five images is acquired at video rate and the temporal smoothing of both, left and right image sequences, starts concurrently to the acquisition. The last two images (on each side) of the time smoothed sequence, are then used to compute the average left and right optical flow vectors. Finally, the difference between the average flow on the left and right images is used to synthesize the control law. Then, a new sequence of 5 images is acquired, and the whole process is repeated. In this way, the images are sampled at video rate even if the complete system operates at a slower rate (determined by the computation time which varies with the number of non-zero flow vectors).

The images acquired have a resolution of 256x256 pixels, and the optical flow is computed over a window of 32x64 pixels on each side. In the current implementation, the averaged optical flow vectors are computed at a frequency of approximately 1.5Hz.

In order to clarify the *Divergent Stereo* approach, we have realized an open-loop experiment, illustrating the obstacle detection capabilities. Figure 2 shows the experiment setup and an image with the flow vectors superimposed. The vehicle is moving at a forward speed of 80mm/s.

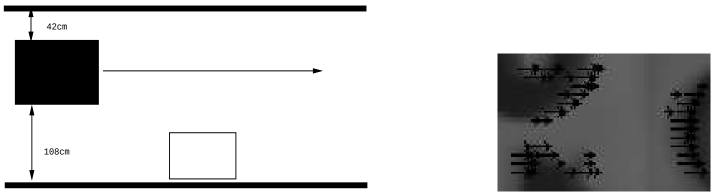

Figure 2: Experimental setup of the obstacle detection experiment. A sample of the computed optical flow is shown on the right.

Along the rectilinear trajectory, the robot will pass midway between the left wall and an obstacle placed on the right side. At this point, the robot is centered relatively to the left wall and the obstacle, and the bilateral flow difference should be 0.

The evolution in time of the difference between the left and right average flow fields is shown in Figure 3. As expected, the difference is initially positive, as the robot is closer to the left than to the right wall and, close to the obstacle, the error approaches zero, as the vehicle is approximately centered with respect to the obstacle and the left wall. After passing the obstacle, the difference tends again to approximately the initial value, as the robot continues to follow a rectilinear path.

Figure 3: Difference between the left and right average flows, during the obstacle detection experiment. The full path takes 20 seconds, covering a distance of about 2.4m. The filtered signal (dashed line) results from applying a first order filter to the measured values.

This experiment shows the sensitivity of the perception process proposed in an obstacle detection task, thus motivating its use in a closed loop fashion. An important remark is that the system does not critically rely on the accuracy of the optical flow computation, because the measurements will continuously be used in closed loop.

3.1 Analysis of rotational flow

The vision-based centering reflex relies on the assumption that the optical flow amplitude is only dependent on the distance between the cameras and the environment. Strictly speaking, this assumption is true if the two cameras are pointing symmetrically with respect to the heading direction, and the heading direction does not change. This constraint does not hold during the rotational motion, necessary to adjust the heading direction, because the roto-translation of the cameras introduces a component on the flow field which does not depend on distance.

This section presents a detailed analysis of the rotational flow of the *Divergent Stereo* setup, and explains how the disturbances introduced during robot rotations, can be reduced and eventually become irrelevant, by appropriately positioning the cameras with respect to the vehicle rotation center and by tuning some of the centering reflex control variables.

Figure 4 describes the set of coordinate systems associated to the robot. Let the robot coordinate system be denoted by {X,Y}, and the left/right cameras coordinate systems by {x,y}. The setup is symmetric and the right camera optic center has polar coordinates (ρ, θ) with respect to the origin of the robot coordinate system.

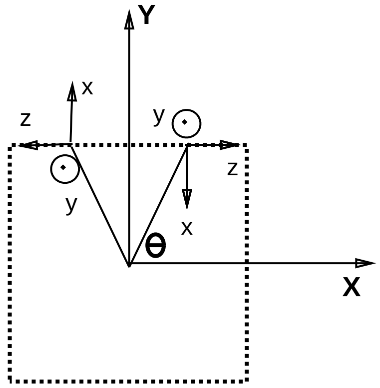

Figure 4: The coordinate systems associated to the vehicle for the analysis of the rotational motion. The robot coordinate system is denoted by {X,Y} and the left/right cameras coordinate systems are denoted by {x,y}. The setup is symmetric and the center of the right camera has polar coordinates (ρ, θ) with respect to the robot coordinate system origin.

For small rotations, first order approximations can be used to determine the translational motion affecting the cameras optic centers:

$$\begin{aligned} \dot{X}_L &= -\rho \, \sin(\theta) \, \dot{\theta} & \qquad \dot{X}_R &= -\rho \, \sin(\theta) \, \dot{\theta} \\ \dot{Y}_L &= -\rho \, \cos(\theta) \, \dot{\theta} & \qquad \dot{Y}_R &= \rho \, \cos(\theta) \, \dot{\theta} \end{aligned}$$

where the subindex refers the left or right camera. By expressing this motion parameters in the cameras coordinate systems, one is led to:

$$\begin{aligned} \omega_y^L &= \dot{\theta} & \qquad \omega_y^R &= \dot{\theta} \\ T_x^L &= -\rho \, \cos(\theta) \, \dot{\theta} + T_M & \qquad T_x^R &= -\rho \, \cos(\theta) \, \dot{\theta} - T_M \\ T_z^L &= \rho \, \sin(\theta) \, \dot{\theta} & \qquad T_z^R &= -\rho \, \sin(\theta) \, \dot{\theta} \end{aligned}$$

where T_M is the robot forward speed.

The influence of the rotation can be perceived in the following equations describing the horizontal component of the image motion field :

$$\begin{aligned} U_L &= \frac{1}{Z_L} \left[x_a \rho \, \sin(\theta) \, \dot{\theta} + \rho \, \cos(\theta) \, \dot{\theta} - T_M \right] - (1 + x_a^2) \, \dot{\theta} \\ U_R &= \frac{1}{Z_R} \left[-x_a \rho \, \sin(\theta) \, \dot{\theta} + \rho \, \cos(\theta) \, \dot{\theta} + T_M \right] - (1 + x_a^2) \, \dot{\theta} \end{aligned}$$

where x_a denotes an image point coordinate expressed in units of focal length.

Observing that the left and right flow fields have opposite directions, (due to the choice of the cameras coordinate systems) the comparison of both left/right lateral flows is given by the sum of U_L and U_R:

$$\begin{aligned} e &= U_L + U_R && (10) \\ &= [T_M - x_a \rho \ \sin(\theta)\,\dot\theta]\left(\frac{1}{Z_R} - \frac{1}{Z_L}\right) + \rho\ \cos(\theta)\,\dot\theta\left(\frac{1}{Z_R} + \frac{1}{Z_L}\right) - 2(1 + x_a^2)\,\dot\theta && (11) \end{aligned}$$

In the absence of rotation, this equation is simplified to :

$$e = T_M\left(\frac{1}{Z_R} - \frac{1}{Z_L}\right). \tag{12}$$

This equation shows that, without rotation, the error signal, e, is directly proportional to the deviation from the center trajectory. The robot forward speed appears as a scaling factor which has the role of a *sensitivity gain*. If the rotational motion is important, the circumstances under which the lateral flow comparison is still meaningful as a trajectory deviation measurement, have to be considered. There are three main contributions to be analyzed :

1. In the first term, $[T_M - x_a\rho\ \sin(\theta)\,\dot\theta](\frac{1}{Z_R} - \frac{1}{Z_L})$, the rotation affects the *sensitivity gain*. Avoiding large variations of this gain, which would influence the system closed-loop behavior, introduces a constraint between the maximum rotation speed and the robot forward speed, T_M. This requirement can be met by suitably selecting the values of θ and ρ, or increasing the vehicle speed.

2. The term $\rho\ \cos(\theta)\,\dot\theta(\frac{1}{Z_R} + \frac{1}{Z_L})$ depends on the unknown 3D structure but does not convey any information on the deviation from the optimal trajectory. It can be made small enough, by installing the cameras closer to the vehicle rotation center, hence reducing ρ.

3. The term $-2(1 + x_a^2)\,\dot\theta$ is not dependent on the 3D structure of the environment. A *calibration* procedure could be designed to compute the correspondence between rotational speed (which is internally controlled) and displacements in the image plane. During normal operation, this term can be compensated.

3.2 Design specifications

The problem of determining a set of design constraints to overcome the problems due to the rotational motion will be analyzed in this section. The analysis will be restricted to the structure dependent terms which appear in equation (11).

Concerning the first term, it can be seen that the rotation leeds to change in the *sensitivity* of the error signal to the trajectory deviation factor (see equation (12)). Very fast rotations could even lead to unstability. Therefore, a natural constraint will be such that the rotation term does not affect the *sensitivity gain*, too significantly.

$$x_a\rho\ \sin(\theta)\,\dot\theta < \alpha T_M \tag{13}$$

where $\alpha \in [0,1]$ is the admissible relative change in the *sensitivity gain*. Alternatively, the second term in the error signal should also be kept small relatively to T_M :

$$\rho \, \cos(\theta) \, \dot{\theta} < \beta \, T_M \tag{14}$$

where $\beta \in [0,1]$ quantifies the admissible relative magnitude of this term.

A suitable setup (ρ, θ) can be chosen to meet these specifications. There are two other ways to put these constraints into practice. The first one, consists in fixing the value for the forward speed, and limiting the value of $\dot{\theta}$, by saturating the control variable before it is applied to the robot (in natural systems this may be an intrinsic constraint):

$$\dot{\theta} < \min \left(\frac{\alpha \, T_M}{x_a \rho \, \sin(\theta)}, \frac{\beta \, T_M}{\rho \, \cos(\theta)} \right). \tag{15}$$

Another way of addressing the problem consists in having an extra control loop, that dynamically adjusts the vehicle forward speed, to fulfill the design constraints:

$$T_M > \max \left(\frac{\dot{\theta} x_a \rho \, \sin(\theta)}{\alpha}, \frac{\dot{\theta} \rho \, \cos(\theta)}{\beta} \right). \tag{16}$$

As a numerical example, let the values of the current setup be considered:

$$\begin{aligned} \theta &= 72^o \\ T_M &= 0.08 \;\; m/s \\ \rho &= 0.34 \;\; m \,. \end{aligned}$$

Since the focal length of the lenses is $4.8mm$, roughly the same order of magnitude as the size of the CCD chip, a reasonable approximation is $x_a \simeq 1$. The following constraints arise from these settings:

$$\dot{\theta} < \min(\, 0.247\alpha, \; 0.759\beta \,) \;\; rad/s \,. \tag{17}$$

For example, by setting the project parameters α and β to 0.5 and 0.2, respectively, yields[5]:

$$\dot{\theta} < 7.1 \;\; deg/s \,. \tag{18}$$

To conclude, one can say that through careful placement of the camera system and some design parameters, the constraints derived in this section can be met. It is worth stressing, however, that the proposed solution only represents an approximation based on qualitative observations. On the other hand, the experiments performed, clearly show that the desired behavior is obtained and that the choice of the design parameters is by no means critical, proving the robustness of the approach proposed.

[5]The choice of these numerical values is a way of specifying a threshold relative to some known nominal quantity.

Obviously, another possibility to overcome the rotation problem is to inhibit the control action whenever the robot is required to rotate faster than the maximum allowable speed (a sort of "saccadic" motion which resembles very much the behavior of insects). During our experiments we have tried several approaches such as using the design specifications explained in this section or adopting the "inhibitory" solution, by actually disregarding the flow measurements during fast rotations. *Robee* performed equally well in both situations.

The solution adopted by Coombs and Roberts is also satisfactory, even if the higher complexity, introduced by the stabilization of the cameras against head rotations and by the system keeping the eyes aligned with the heading direction, does not seem to improve behavioral performance.

4 Real-time Control

The overall structure of *robee* control system involves two main control loops:

1. Navigation loop - Controls the robot rotation speed in order to balance left and right optical flow.

2. Velocity loop - Controls the robot forward speed by accelerating/decelerating as a function of the amplitude of the lateral flow fields[6].

Additionally, a *sustaining mechanism* is implicitly embodied in the control loops to avoid erratic behaviors of the robot, as a consequence of localized (in space and time) absence of flow information. These aspects and the analysis of the different control loops will be presented in the following sections.

4.1 Navigation Loop

The analysis of the control loops is based on simplified dynamical models of the control chain components and on the use of linear systems theory[7].

The robot heading direction (*controlled variable*) is controlled by applying a rotation speed (*control variable*) superimposed on the forward motion. The simplest dynamic model of the system must account for two important terms: an integral term relating the input angular speed and the output angular position; the mechanical inertia of the system, which is modeled by a first order dynamic system. In continuous time, the transfer function relating the *control* and *controlled* variables, according to our simple model, is given by:

$$G_c(s) = \frac{a}{s\,(s+a)} \tag{19}$$

[6] *Robee* accelerates if the lateral flow is small (meaning that the walls are far away), and slows down whenever the flow becomes larger (meaning that it is navigating in a narrow environment).

[7] A more accurate control analysis/synthesis is undergoing.

where a is the dominant low-frequency mechanical pole.

Since we are using digital control, we must determine how the computer (where the control algorithm is implemented) "sees" the system. As a sample-and-hold mechanism is being used, the *step invariant* method is appropriate to determine the discretized system [16]. Considering the low-frequency pole at 5 Hz, and a sampling period of $\tau = 0.7s$, we obtain:

$$G_d(z) = \frac{0.468z + 0.0318}{z^2 - (1 + 1.51e^{-7})z + 1.51e^{-7}} \, . \tag{20}$$

Again for simplicity, we may assume that the difference between the left and right flow vectors, provides the *error* between the robot direction and the direction of the lateral walls, with a delay of one sampling period introduced by the flow computation. This approximation is only valid for small course deviations, since the visual process is, in fact, non linear. The sensor model is then given by:

$$e(k) = \theta_{walls}(k-1) - \theta_{robot}(k-1) \, . \tag{21}$$

Qualitatively, $e(k)$ is positive if the left side flow is larger than the right side flow, which means that the left wall is closer than the right one. Hence, the appropriate control action would be turning to the right. The discrete time PID controller, that is used to close the navigation loop, performs the following control law:

$$u(k) = K_p[\ e(k) + K_i \sum_n e(k-n) + K_d(e(k) - e(k-1))\ ] \tag{22}$$

where u is the control variable (rotation speed in degrees/s) and $e(k)$ stands for the *error signal* observed at time instant k. The transfer function corresponding to the PID is given by:

$$G_{PID}(z) = K_p \frac{(K_i + K_d + 1)z^2 - (1 + 2K_d)z + K_d}{z(z-1)} \, . \tag{23}$$

By connecting all these models, we obtain a linear feedback loop as shown in Figure 5.

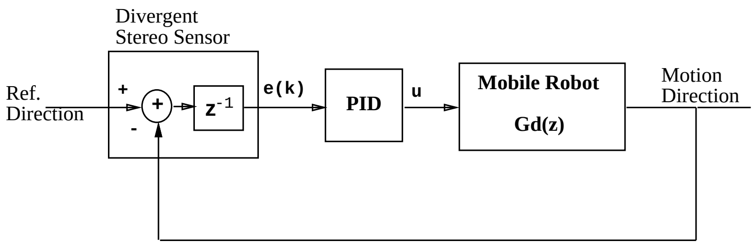

Figure 5: This block diagram illustrates a simple modelization of the robot navigation system

Although a thorough analysis of the system behavior, with different control settings, can hardly be done based on approximate models, a discussion can still provide us some insight and understanding on the performance of several classes of controllers.

As a start, the loop could be closed simply by using a proportional gain (by setting the integral and derivative gains to zero). The root-locus corresponding to this situation is shown in Figure 6. Even though the proportional controller succeeds in stabilizing the system, fast responses can only be attainable with large values of gain which lead to significant oscillatory behavior, as the dominant poles become complex conjugate.

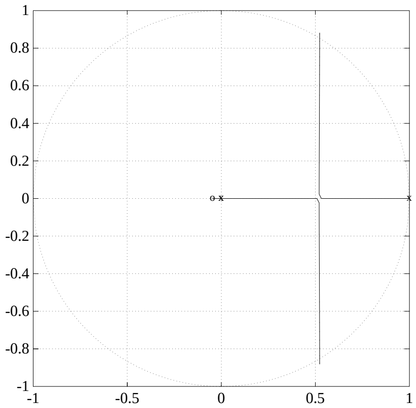

Figure 6: Root-Locus corresponding to the proportional controller. The unit circle is shown in dotted line for easier stability analysis.

By adding a derivative component (which basically works as a predictive term, thus coping better with the delay) to the controller, one can expect to achieve faster responses, even for smaller gain values. Figure 7 shows the root-locus in this situation. The derivative component is fixed to $K_d = 1.5$. The effect of adding the extra zero is that the low frequency pole is attracted into the higher frequencies, thus improving the response time of the system. Also, for large gains the oscillatory behavior of the system is reduced.

Finally, the effect of inserting the integral component of the controller can be analyzed by considering the root-locus shown in Figure 8. The parameters used are $K_i = 0.1$ and $K_d = 1.5$. In this last situation, adding a discrete integrating effect (an extra pole in 1), decreases the stability margin of the system, as well as the response speed. In practice, the system may even become unstable due to various unmodeled components or perturbations, such as the non linearities in the visual processing.

This simplified analysis, allowed us to gain important insight and understanding on the behavior of the system under the different controller topologies. Further modeling is still necessary for a quantitative analysis or a more systematic control synthesis. Nevertheless, some of the ideas discussed in this section were later on verified in the experiments, as described in Section 5.

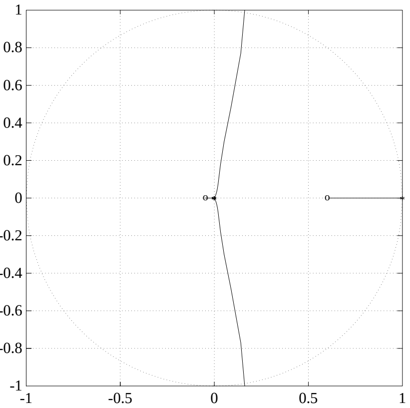

Figure 7: Root-Locus corresponding to the proportional-derivative controller. The derivative gain was fixed at $K_d = 1.5$.

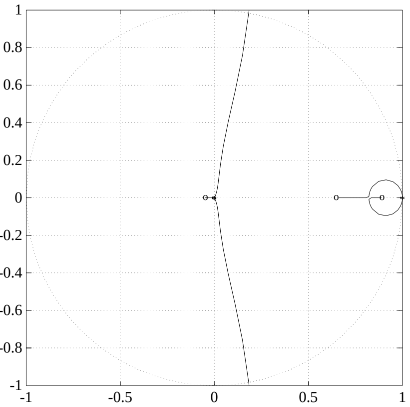

Figure 8: Root-Locus corresponding to the PID controller.

4.2 Velocity Control

This section presents the strategy proposed to control the robot forward speed based on the environment structure. The rationale is that, if the robot is navigating on a narrow environment, it is safer to decrease the forward speed; whereas if it is moving on a wide clear area, it is reasonable to increase its speed.

The mean flow vector on each side of the robot gives a qualitative[8] measurement of depth. By averaging these bilateral mean flow vectors, a qualitative measurement of width is obtained.

The control objective amounts to keeping the average flow close to some specified reference value. If the flow increases, the robot should slow down, thus reducing the observed flow. This behavior, which can be implemented within the purposive approach described in this paper, is not only coherent from the perceptual viewpoint (it agrees with what a human driver, for example, would do) but also increases the safety. Qualitatively, this corresponds to saying that the size of the environment is scaled by the robot speed.

Let T_o be the nominal speed, at which the robot should move, and let f_o be the corresponding nominal flow. For safety purposes, the robot speed, T, is constrained to the interval $[(1-\beta)T_o,\ (1+\beta)T_o]$, where $\beta \in [0,1]$ quantifies the permissible excursion. A sigmoid function is used as smooth saturation, as shown in Figure 9.

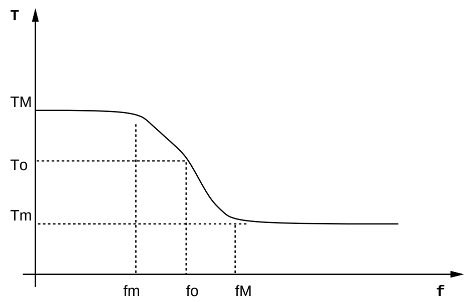

Figure 9: The sigmoid function, used in the velocity control loop, describes how the robot forward speed should change, with relation to the measured flow. It is used to introduce a smooth saturation on the robot velocity.

The velocity to be applied to the robot is given by:

$$T = T_o \left[1 - \beta + \frac{2\beta}{1 + e^{\delta(f - f_o)}} \right] \tag{24}$$

where f is the average between the left and right flows, and δ determines how fast should the speed variation be with respect to the flow variation. To determine δ, one can say, for example, that 90% of the total velocity excursion should be reached for a relative variation of ϕ around

[8]Qualitative in the sense that it is not a measure of depth, and, moreover, depends on the robot speed.

the nominal flow. By using equation (24), δ is given by:

$$\delta = \frac{\ln 19}{\phi f_o} \,. \tag{25}$$

In the current implementation, we have used $\beta = 0.5$, $\phi = 0.3$, and a reference flow of $f_o = 2.0$ pixels/frame, yielding $\delta = 4.95$.

4.3 "Sustained" behavior

The navigation system, described in the previous sections, allows the robot to navigate by balancing the flow measurements on the left and right sides. Therefore, it can only be applied as long as there is texture on both sides of a corridor–like environment. This situation is not entirely satisfactory for two reasons:

- It would be nice if the reactive behavior of *Robee* could be used in environments far more complex than corridors.
- In most mission scenarios, *Robee* would most certainly find environments with "walls" not uniformly covered with texture. This fact would cause an illusory perception of infinite distance and, therefore, elicit unappropriate behaviors (for example, if an open door is found while traveling along a corridor).

To overcome these problems, we have introduced in the control strategy a mechanism able to cope with unilateral lack of flow information. Whenever it occurs, the control system uses a reference flow that should be sustained on the "seeing" camera (i.e. the camera still measuring reliable flow).

This mechanism monitors whether there is significant flow being measured on both sides of the vehicle, or if just one (or none) of the cameras is capturing significant flow. Three situations may arise:

- **Bilateral flow** - Optical flow is measured in both cameras. Consequently, the robot is locally navigating in a corridor, and the standard navigation strategy can be applied
- **Unilateral flow** - Only one camera is capturing flow information. Without the *sustaining* mechanism, the robot would simply turn towards the side without flow measurements, trying to balance lateral flows. A more appropriate behavior, instead, would be keeping the unilateral flow constant, hence following the ipsilateral wall at a fixed distance. With such strategy, the robot may cross corridors with open doors, even with partially untextured walls and, when arriving to a room, follow the walls at a fixed distance.

- **Blind -** If none of the cameras is capturing flow information, the robot is virtually blind. This robot should either stop or follow straight ahead for a while, or even wander in a random exploratory way, until some texture is again found. Currently, *robee* stops if this situation arises.

Let us analyze how, without any prior knowledge of the environment, the *sustaining* mechanism is implemented . During normal operation, the reference flows are estimated by filtering over time each of the lateral mean flow vectors. The time filtering takes into account the number of vectors that contributed to the mean computation :

$$\bar{f}_{(t)} = \frac{\alpha \bar{n}_{(t-1)} \bar{f}_{(t-1)} + (1-\alpha) n_{(t)} f_{(t)}}{\alpha \bar{n}_{(t-1)} + (1-\alpha) n_{(t)}} \tag{26}$$

$$\bar{n}_{(t)} = \frac{\alpha \bar{n}^2_{(t-1)} + (1-\alpha) n^2_{(t)}}{\alpha \bar{n}_{(t-1)} + (1-\alpha) n_{(t)}} \tag{27}$$

where $f_{(t)}$ is the mean flow, computed at time t, with $n_{(t)}$ flow vectors; $\bar{f}_{(t)}$, $\bar{n}_{(t)}$ are the corresponding *time filtered* values; and $\alpha \in [0, 1]$ is a time decay constant. Functionally, α determines the amount of past flow information that should be "remembered" in the filtering process.

The system enters the *sustaining mode* whenever it is unable to estimate reliable flow vectors on one side. In the experiments realized, α was set to 0.6, which agrees with the time filtering parameter used for temporal smoothing.

As a final remark, we would like to stress that the approach proposed extends considerably the performance of the "reactive" behavior and that the use of both the *corridor* and the *wall* following behaviors in task–driven navigation is straightforward. In particular, two potential applications are worth mentioning:

- the "reactive" control of *robee* can be used to acquire information about environmental structure. In fact, odometric information, coupled with the sensory information of *robee*, would allow to build a map of the environment in terms of *corridors* and *walls* which could be used by a planning system to drive navigation on a higher level.

- The planner of a robot moving in a known (but variable) environment could take advantage of "task–level" commands like: "navigate to the end of the corridor" or "follow a wall" without relying on geometric information which, ultimately, may be difficult to acquire and maintain in realistic environments.

5 Results

This section presents a set of tests that clearly demonstrate several applications of the *Divergent Stereo* navigation approach. The goal of the experiments is the study of the closed loop behavior of the visually guided robot. In order to test the performance in a wide range of environmental situations and to analyze in more detail the influence of the different controller settings, three experimental situations with increasing degree of difficulty have been considered. The final experiment illustrates the *sustaining* mechanism along with the velocity control loop. In all the results presented the trajectory of the robot was recorded from odometric data during real-time experiments.

5.1 Turn Experiment

On the first set of experiments, the robot was tested in a turning corridor setup, as shown in Figure 10.

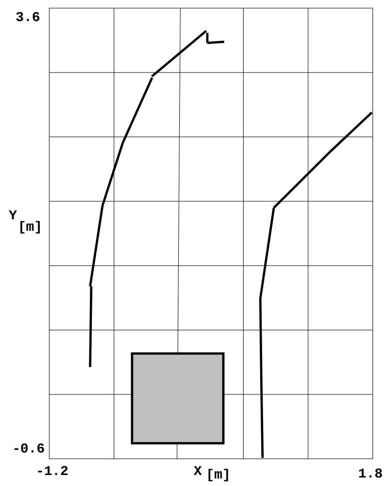

Figure 10: Setup used for the *turn* experiment. The vehicle is supposed to perform the turn, using the divergent stereo navigation strategy.

The whole navigation system has been tested with different controller settings. For these experiments, the integral gain K_i, was fixed to zero, as suggested by the discussion on the controller design.

In the first three experiments, the influence of the derivative gain on the navigation performance has been studied. Figure 11, shows the trajectories followed by the robot with a fixed proportional gain, $K_p = 1.5$, and using for K_d the values of 1.0, 1.2 and 1.4. The trajectories were recorded using the odometry information, and are shown superimposed on the experimental setup.

To test the effect of the proportional gain on the controller, another set of trials was performed. The integral and derivative gains were kept constant ($K_i = 0$ and $K_d = 1.2$), while using for K_p the values of 1.25, 1.5 and 1.75. The results are shown in Figure 12.

The analysis of the trajectories show that, by increasing the value of K_d, the response becomes faster, even though the trajectories may become less smooth. On the other hand, by

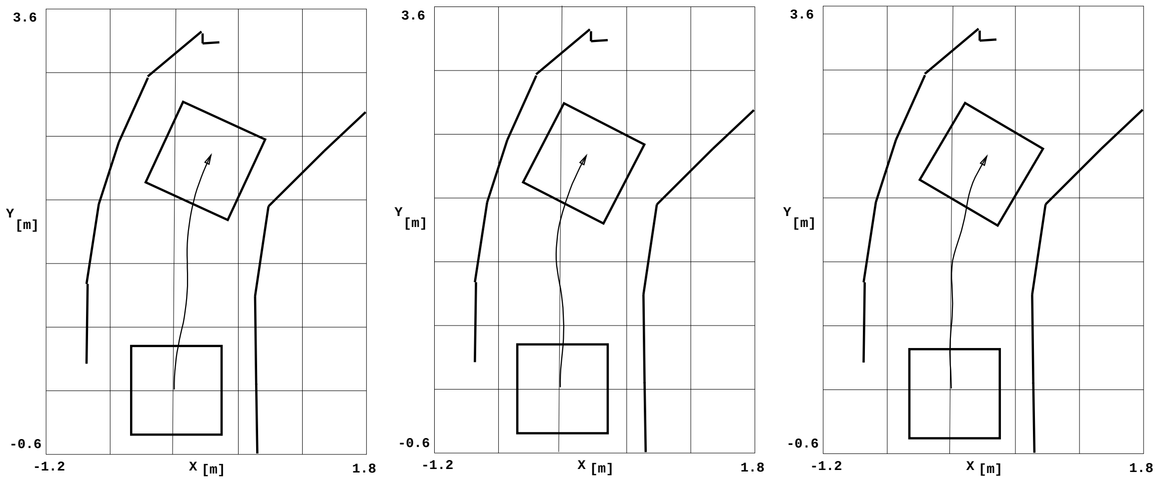

Figure 11: Results obtained in the closed loop operation in the turn experiment. The controller settings were $K_p = 1.5$, $K_i = 0$ and using for K_d the values (from left to right) of 1.0, 1.2 and 1.4. The trajectory was recovered using odometry.

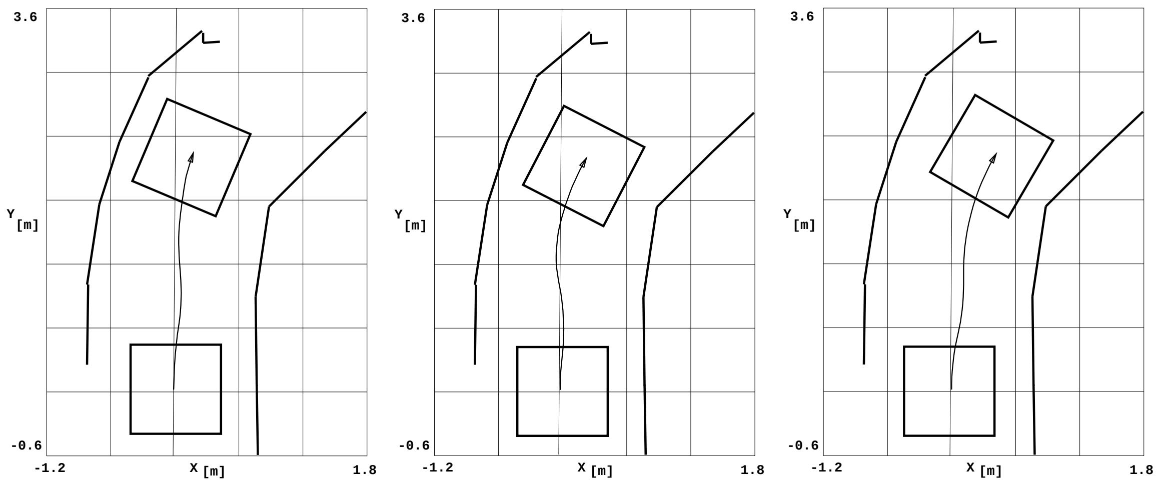

Figure 12: Results obtained in the closed loop operation in the turn experiment. The controller settings were $K_d = 1.2$, $K_i = 0$ and using for K_p the values (from left to right) of 1.25, 1.5 and 1.75. The trajectory was recovered using odometry.

increasing K_p, the response becomes faster but the vehicle does not succeed to have such fast reactions as by increasing the derivative component.

5.2 Funnel

In another set of experiments, we used a funneled corridor with an obstacle. The setup is shown in Figure 13. When navigating in this environment, the robot must avoid the obstacle while trying to keep centered in the funneled corridor. This situation is particularly interesting because it forces the robot to react to sudden changes (the obstacle) as well as to smooth changes of the environment structure.

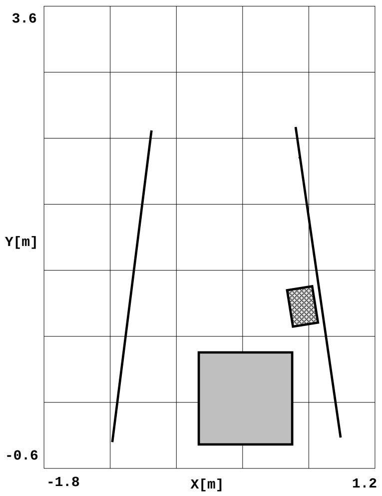

Figure 13: Setup used for the funneled corridor experiment. In this scenario, the robot has to avoid an obstacle while managing to keep the track in the funneled corridor.

Again, different settings of the PID controller have been used, in order to study the robot behavior. Figure 14 shows the trajectories corresponding to three of those experiments. The robot trajectory, recovered from odometry, is superimposed on the setup layout.

The first trial represents an experiment with the controller tuned with $\{K_p = 1.5$, $K_i = 0.0$, and $K_d = 1.5\}$, which led to a nice trajectory. On the second trial, the integral component has been introduced using $\{K_p = 1.5$, $K_i = 0.1$, $K_d = 1.5$ $\}$. As suggested by the discussion made in Section 4, an unstable behavior was observed, with the vehicle moving towards the left wall. Finally, on the third experiment a nice performance was also obtained by slightly increasing the proportional gain, while diminishing the derivative gain $\{K_p = 1.8$, $K_i = 0.0$, $K_d = 1.2\}$. As expected, a somewhat smooth trajectory was obtained.

The tests performed, enhance the importance of suitable control system design, and clearly show the use of our discussion of the modeling and control system design. Even if further modeling is, certainly, necessary it is worth noting that the behavior of the robot does not

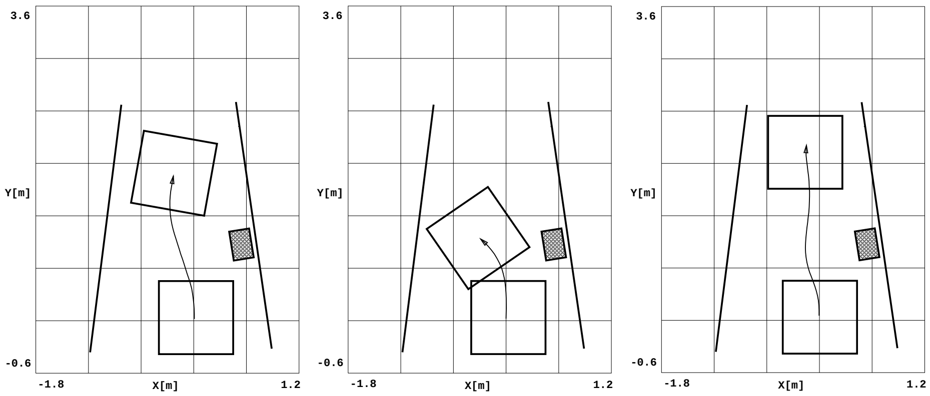

Figure 14: Funneled corridor experiment. The leftmost and rightmost plots show the results of increasing the proportional gain while decreasing the derivative gain. In the center, it is seen the unstable behavior due to the insertion of the integral action.

depend upon critical values of the controller settings and, therefore, the robustness of the system is very promising.

5.3 Corridor

In this set of experiments, a corridor which is just slightly larger than the robot and with a sharp turn in the end has been used (see Figure 15) to test the performance of the system on a combination of different environmental situations.

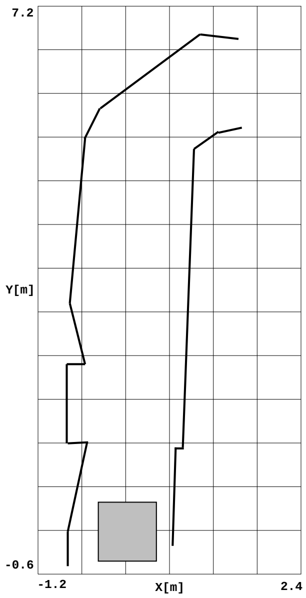

Figure 15: Setup used for the corridor experiment. The corridor is just slightly larger than the robot, and has a tight turn in the end.

Also in this situation, several trials were performed, changing the parameters of the PID controller. Figure 16 shows the trajectories obtained by increasing the values of K_p. The derivative gain is fixed in $K_d = 1.2$, while the proportional gain increases from left to right $K_p = 1.2,\ 1.4,\ 1.6$. The integral gain is not used.

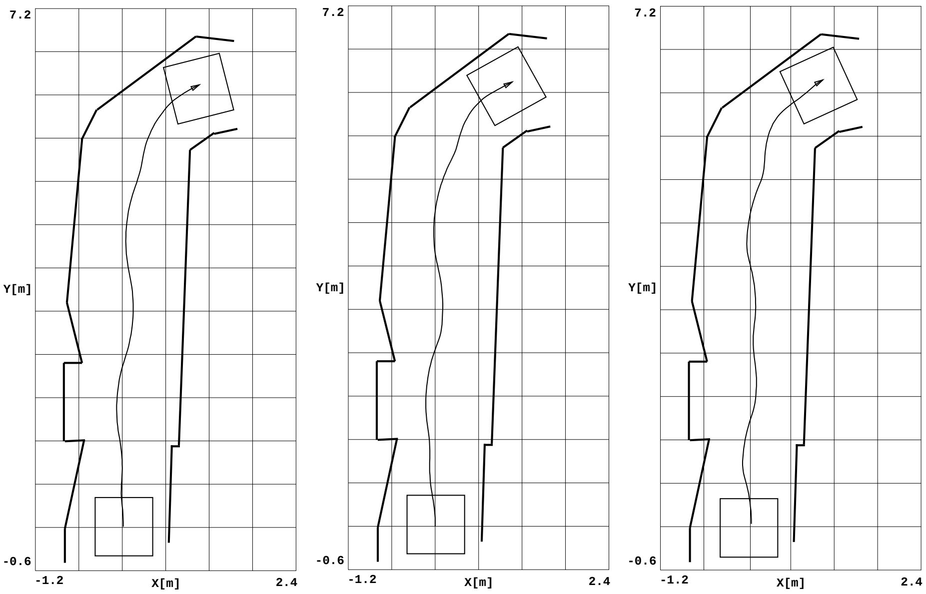

Figure 16: Corridor experiment. The derivative gain is fixed in $K_d = 1.2$, while the proportional gain increases from left to right $K_p = 1.2, 1.4, 1.6$. The integral gain is not used.

On the second set of trials, the proportional gain is set to $K_p = 1.4$, and the derivative gain is increased from left to right $K_d = 1.0,\ 1.2,\ 1.4$. Again, the integral gain is not used. The trajectories are shown in Figure 17.

The results show that by increasing, both the proportional and derivative gains, the response becomes somewhat faster. In general, by increasing the derivative gain leads to a faster behavior of the control system.

5.4 Velocity Control

In order to test the velocity control, the robot was navigating through a funneled corridor, where the width changes from 1.65m to 1.25m. The full length of the funneled corridor is about 2.25m.

Since the corridor is becoming narrower, the average flow increases. By introducing the velocity control mode, the robot velocity will decrease in order to keep a constant flow. Figure 18 shows the average between the left and right flows measured over time. The filled line was obtained without velocity control (showing increasing flow values), while the dotted line shows the action of the velocity control keeping image flow close to the desired value of 2 pixels/frame.

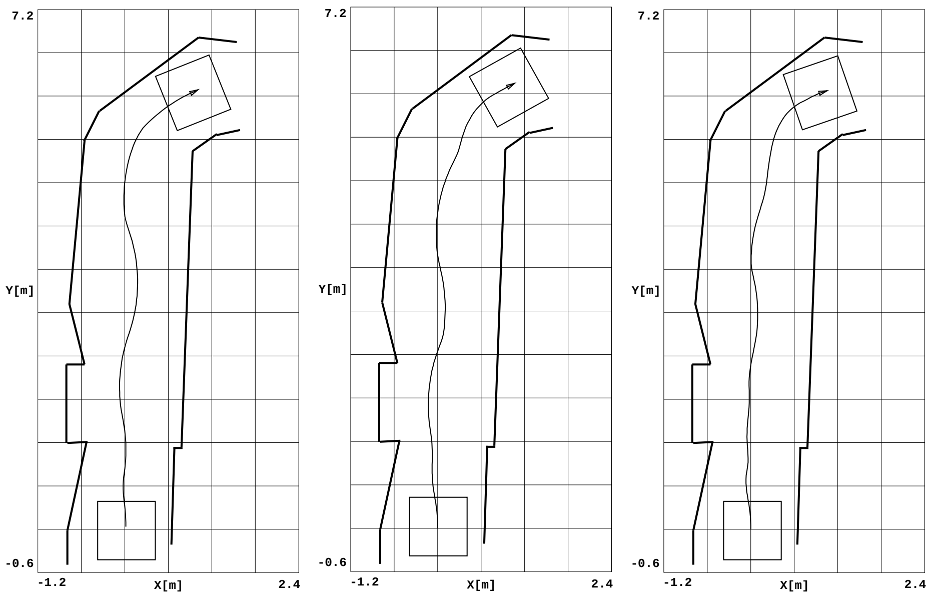

Figure 17: Corridor experiment. The proportional gain is fixed in $K_p = 1.4$, while the derivative gain increases from left to right $K_d = 1.0, 1.2, 1.4$.

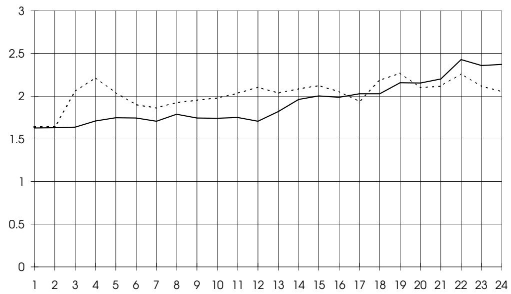

Figure 18: The average between the left and right flows obtained in an experiment with (dotted line) and without (filled line) velocity control. The nominal flow is 2.0 pixels/frame and the nominal velocity 80mm/s.

The evolution of the robot velocity along the path is shown in Figure 19. It is seen that, as the corridor is narrowing, the velocity decreases in order to keep the flow to a lower safer value.

Other experiments were performed using the proposed strategy. Particularly, it is interesting to see that, during the corridor experiment, the final turn is done at a reduced speed enabling the robot to make a softer, safer turn.

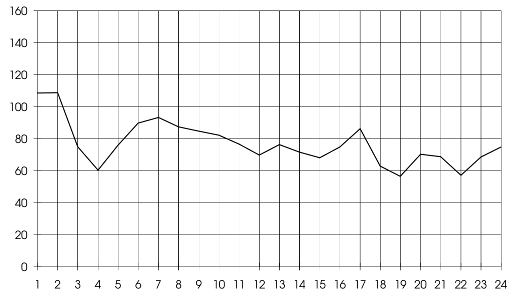

Figure 19: The robot speed in mm/s during the funneled corridor experiment. As the corridor narrows, also the velocity decreases for safer operation.

5.5 Sustained behavior

This experiment was designed to show the influence of the sustaining mechanism upon the robot behavior and how it can deal with different environment structures. The first experiment consisted in using a corridor which exhibits a lack of texture on the right side, and whose left wall suddenly finishes in a room. Figure 20 shows the trajectories obtained by activating (center) and de-activating (left) the sustaining mechanism. In the latter case, the robot turns left trying to balance both lateral flows while, in the former, a reference flow value is sustained leading the robot to follow the right wall.

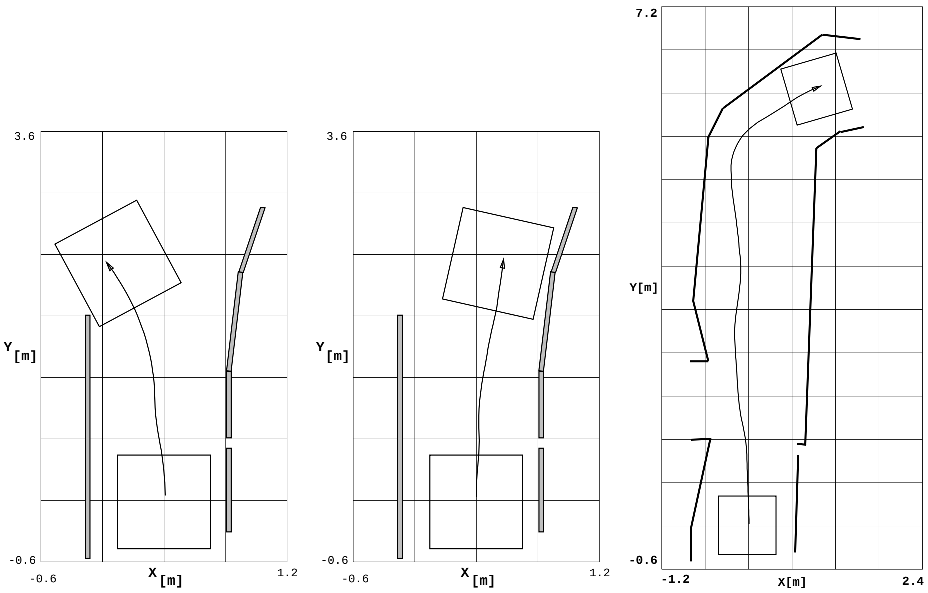

Figure 20: Sustained mechanism experiment. The left diagram corresponds to the behavior of the robot without the sustaining mechanism. At the center, due to the sustaining behavior, the robot manages to follow the right wall. The rightmost diagram corresponds to an experiment along the corridor with an open door on the left and a lack of texture on the right.

Another experiment was made using the corridor setup, where some texture was removed from the corridor walls, and a door, roughly located at the corridor center, is left opened. The results are documented in Figure 20.

To conclude, it seems reasonable to claim that the proposed approach, both on the control and visual perception facets, led to good results and proved the feasibility of a navigation system based on these principles. Furthermore, it should be noticed that with the introduction of the sustained behavior, the robot is able to navigate in a much wider set of environments. In fact, only one textured wall is needed for the navigation strategy to work.

It is worth noting that, the design principles adopted to eliminate the influence of the rotational component of the optical flow, did prove successful in our experimental conditions and, therefore, even if a stabilization mechanism is desirable or even mandatory in other situations, it is not strictly necessary to produce the kind of behaviors described in this paper.

6 Conclusions

A qualitative approach to visually-guided navigation based on optical flow has been presented motivated by studies and experiments performed on freely flying honeybees.

The approach is based on the use of two cameras mounted on a mobile robot with the optical axes directed in opposite directions such that the two visual fields do not overlap (*Divergent Stereo*). Range is perceived by computing the apparent image speed on images acquired during robot motion.

A real-time computation of optical flow is presented, based upon the constraints imposed by the geometry of the cameras and by the navigation strategy. Furthermore, some suggestions have been presented on how to select some design variables in order to make the disturbances due to the rotational motion, irrelevant to the reactive behaviors described in the paper.

A PID controller was used to close the visuo-motor control loop. The closed loop behavior was studied, based on models of the different control system components. The analysis of the control system design led to a suitable configuration for the PID controller.

The approach has been tested using real-time experiments to accomplish different navigation tasks, like performing a tight turn or navigating through a funneled corridor. The influence of the control parameters on the system behavior was studied and the results confirmed, within the assumptions made, the discussion on the control system design.

A controller for the robot forward velocity was also studied and implemented. Experiments have been made to show the improvement achievable, by including this control loop in cluttered environments.

Finally, through the insertion of a sustained behavior, the robot is able to operate in environments rather more complex than a simple corridor, showing the capability of operating in sparsely textured corridors, and following unilaterally textured walls.

All the experiments were performed without the need for accurate depth or motion estimation, nor required a calibration procedure (besides the manual positioning of the two cameras).

The main features of Robee can be summarized as follows:

- **Purposive** definition of the sensory apparatus and the associated processing. The approach proposed, in fact, cannot be considered general but, with limited complexity, solves a relevant problem in navigation: the control of heading direction in a cluttered environment.

- Use of **qualitative and direct** visual measures. In our opinion this is not only a "religious" issue but, more importantly, a way to achieve a reasonable autonomy with limited computational power. Successful examples of this approach have recently appeared in the literature both with respect to reflex-like behaviors for obstacle avoidance [17, 18, 19, 20] and in relation to more "global" measures of purposive navigation parameters [21].

- **Continuous use of visual measures**. A further aspect worth mentioning is the attempt made at developing a sensory system providing a continuous stream of environmental information. A first advantage is the increased robustness implicit in the use of repeated measures (no single mistake produces catastrophic errors) and a secondary, and potentially more important, advantage is the possibility of implementing sensory-motor strategies where the need for a continuous motor control is not bounded by an "intermittent" flow of sensory information. This paradigm is, in our opinion, a non trivial evolution of some active vision implementations where the motion of the (active) observer is seen "only" as a way of taking advantage of the stability of the environment (e.g. by moving the vehicle along pre-programmed, known trajectories to reduce uncertainty). The use of vision **during** action [22], on the contrary, may be a very powerful extension of the concept of active observer by exploiting the use of dynamic visual information not only at the "reflexive" level of motor control.

- **Simplicity**. This feature is often regarded as an engineering and implementation aspect and, as such, is not explicitly considered a scientific issue. This view, in our opinion, must be changed if reasonable applications of computer vision are addressed. The issue of simplicity, however, should not be considered, within specific aspects of intelligent actor's design (such as, sensory systems, mechanical design, computational architecture etc.) but must be considered at system level. Robee is an example of such an holistic view of simplicity where the purpose is achieved by a comprehensive analysis and integration of visual processing, sensor design, sensor placement, control law and vehicle structure. In this respect low-level animals (and insects in particular) are extremely interesting examples of "simple" actors where all engineering aspects are mixed exploiting not only "computational" issues but, more importantly, the cooperation of "intelligent" solutions which, if considered separately, may look like interesting implementational tricks but, once acting together, produce intelligent behaviors.

Before concluding, it is worth mentioning the fact that, at its current level of implementation, *robee* is not entirely satisfactory (even within the intrinsic limitations of a reflex-like control) because it is blind in the direction of motion (it will bump against an obstacle just in front of it). On the other hand we are currently using only a very limited portion of the visual field. The use of a more frontal part of the visual field, is currently investigated to extract other motion-derived measures (e.g. time-to-crash) which could help *robee* not only to implement the behaviors described in this paper, but also to control docking speed.

References

[1] N. Franceschini, J. Pichon, and C. Blanes. Real time visuomotor control: from flies to robots. In *Fifth Int. Conference on Advanced Robotics*, Pisa, Italy, June 1991.

[2] D. Coombs and K. Roberts. Centering behaviour using peripheral vision. In D.P. Casasent, editor, *Intelligent Robots and Computer Vision XI: Algorithms, Techniques a nd Active Vision*, pages 714–21. SPIE, Vol. 1825, Nov. 1992.

[3] G. Sandini, J. Santos-Victor, F. Curotto, and S. Garibaldi. Robotic bees. Technical report, LIRA-Lab - University of Genova, October 1992.

[4] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi. Divergent stereo for robot navigation: Learning from bees. In *Proc. CVPR-93*, New Yors, U.S.A., 1993.

[5] M.V. Srinivasan, M. Lehrer, W.H. Kirchner, and S.W. Zhang. Range perception through apparent image speed in freely flying honeybees. *Visual Neuroscience*, 6:519–535, 1991.

[6] R. K. Bajcsy. Active perception vs passive perception. In *Proc. Third IEEE Workshop on Computer Vision: Representation and Control*, pages 13–16, Bellaire (MI), 1985.

[7] D.H. Ballard, R.C. Nelson, and B. Yamauchi. Animate vision. *Optics News*, 15(5):17–25, 1989.

[8] J. Aloimonos. Purposive and qualitative active vision. In *Proc. of Int. Workshop on Active Control in Visual Perception*, Antibes (France), 1990.

[9] G. Horridge. The evolution of visual processing and the construction of seeing systems. *Proc. Royal Soc. London*, pages 279–292, 1987.

[10] M. Lehrer, M.V. Srinivasan, S.W. Zhang, and G.A. Horridge. Motion cues provide the bee's visual world with a third dimension. *Nature*, 332 No.6162:356–357, 1988.

[11] M.V. Srinivasan. Distance perception in insects. *Current Directions in Psychological Science*, 1:22–26, 1992.

[12] B. K. P. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17 No.1-3:185–204, 1981.

[13] Nagel H. On the estimation of optical flow: Relations between different approaches and some new results. *Artificial Intelligence*, 33:299–323, 1987.

[14] S. Uras, F. Girosi, A. Verri, and V. Torre. Computational approach to motion perception. *Biological Cybernetics*, 60:69–87, 1988.

[15] E. DeMicheli, G. Sandini, M. Tistarelli, and V. Torre. Estimation of visual motion and 3d motion parameters from singular points. In *Proc. of IEEE Int. Workshop on Intelligent RObots and Systems*, Tokyo, Japan, 1988.

[16] K. Astrom and B. Wittenmark. *Computer Controlled Systems: Theory and design.* Prentice-Hall, 1986.

[17] G. Sandini and M. Tistarelli. *Robust Obstacle Detection Using Optical Flow.* Proc. of IEEE Intl. Workshop on Robust Computer Vision, Seattle, (WA), Oct. 1-3, 1990.

[18] W. Enkelmann. Obstacle detection by evaluation of optical flow fields from image sequences. In *Proc. of first European Conference on Computer Vision*, pages 134–138, Antibes (France), 1990. Springer Verlag.

[19] F. Ferrari, M. Fossa, E. Grosso, M. Magrassi, and G. Sandini. A pratical implementation of a multilevel architecture for vision-based navigation. In *Proceedings of Fifth International Conference on Advanced Robotics*, pages 1092–1098, Pisa, Italy, June 1991.

[20] M. Fossa, E. Grosso, F. Ferrari, G. Sandini, and M. Zapendouski. A visually guided mobile robot acting in indoor environments. In *Proc. of IEEE Workshop on applications of Computer Vision*, Palm Springs, U.S.A., 1992.

[21] Cornelia Fermüller. Navigation preliminaries. In Yiannis Aloimonos, editor, *Active Perception*, pages 103–150. Lawrennce Erlbaum Associates, 1993.

[22] G. Sandini, F. Gandolfo, E. Grosso, and M. Tistarelli. Vision during action. In Yiannis Aloimonos, editor, *Active Perception*, pages 151–190. Lawrennce Erlbaum Associates, 1993.