Simulating *Pursuit* with Machines

Experiments with Robots and Artificial Vision

J. Dias, C. Paredes, I. Fonseca, H. Araújo, J. Batista, A. de Almeida
Instituto de Sistemas e Robótica
Departamento de Engenharia Electrotécnica
Largo Marques de Pombal - Universidade de Coimbra
3000 COIMBRA, PORTUGAL

Abstract

This article is concerned with the simulation of pursuit. The article describes one solution for the problem of pursuit by using a mobile robot and an active vision system. The solution deals with the interaction of different control systems using visual feedback and it is accomplished by the implementation of a visual gaze holding process interacting cooperatively with the control of the trajectory of a mobile robot. These two systems are integrated to follow a moving object at constant distance and orientation with respect to the mobile robot. The orientation and the position of the active vision system running a gaze holding process give the feedback signals to the control used to pursuit the target in real-time. The mechanisms of cooperation between the different control and visual algorithms are described. The final solution is a system able to operate at approximately human walking rates.

1 Introduction

The pursuit of moving objects with machines such as one mobile robot equipped with an active vision system deals with the problem of integration and cooperation between different systems. This integration has two distinct aspects: the interaction and cooperation between different control systems and the use of a common feedback information provided by the vision system. In this article a global solution is proposed based on the visual *gaze holding* process to establish a pursuit mechanism for *targets* moving in front of the mobile robot. The system is controlled to maintain constant the distance and the orientation of the robot and the vision system.

The solution for this problem deals with the interaction of different control systems using visual feedback. It also addresses the real-time tracking of objects by using a vision system. This problem has been addressed in different fields such as surveillance, automated guidance systems and robotics in general. Several works addressed the problems of visual servoing but they are mainly concerned in object tracking by using vision and manipulators [1], [3], [7], [8], [10].There are also other results reported towards the use of vision information for tracking. For example, Burt reports a real-time feature detection algorithm using hierarchical scaling images for surveillance and robotics [5].

These studies have connection with the solution for pursuit proposed in this article, since they deal with the tracking problem also using visual information. However in our system the *distance* and *orientation* of the mobile robot is maintained with respect to a *target*. The solution includes the control of the *gaze* of the active vision system. Furthermore, our hierarchical control scheme establishes a pursuit process using different degrees of freedom on the active vision system and the movement of the mobile robot.

2 *Pursuit* of Moving Objects

The problem of pursuing a moving object is essentially a motion matching problem. The machine, the robot in our case, must be controlled to reach the same motion as the *target*. In practice this is equivalent to maintain constant the distance and orientation from the robot to the *target*.

To perform the pursuit of a moving *target* (see figure 1), we use two basic control schemes: a visual *fixation* control and the trajectory control of the robot. The visual *fixation* control guarantees that the *target* is continuously tracked by the vision system, and gives information about its position to the robot control.

IEEE International Conference
on Robotics and Automation
0-7803-1965-6/95 $4.00 ©1995 IEEE

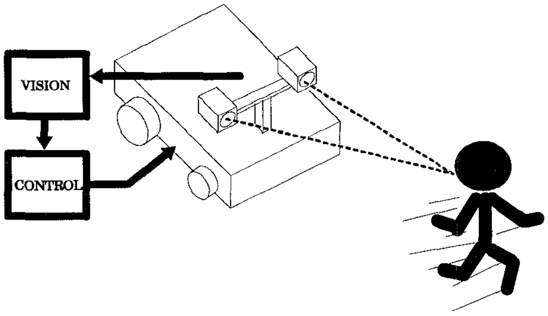

Figure 1: The information provided by the *active vision system* is used to control the mobile robot for *pursuit* a target in $real - time$.

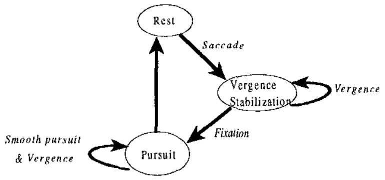

Figure 2: State diagram of the *pursuit* process.

The robot control uses that information as a feedback to maintain the distance and orientation to the *target.*

The visual *fixation* control must be one visual process that runs in the active vision system and has capabilities to define a *target*, to concentrate the vision system on the *target* and follow it. A process with these characteristics has similarities with the visual gaze-shifting mechanism in the humans [6]. The *fixation* centers and holds the orientation of the vision system on a point in the environment. The principle is described graphically by the figure 1 where the mobile robot with an active vision system is concentrated on person. *Fixation* gives a useful mechanism to maintain the relative orientation and translation between the referential in the vehicle and the *target* that is followed.

The pursuit process can be described graphically by the state diagram in figure 2. If a *target* is detected, a *saccade* movement is initialized to put the *target* in the image center, and the system changes from the state *Rest* to *Vergence Stabilization.* During the *saccade* movement no visual information is processed. During the *Vergence Stabilization* the system adjusts its *fixation* in the *target.* This is equivalent to establishing the correct correspondence between the centers of the two images, defining a *fixation* point in the *target.* Since the vergence is stabilized, the system is maintained in the *Pursuit* state.

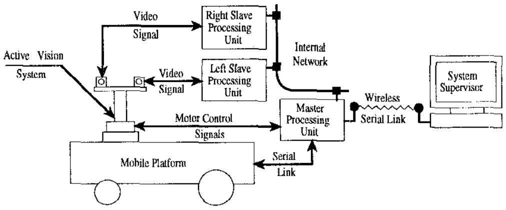

Figure 3: System Architecture.

3 Building a pursuit system

3.1 System architecture

The main hardware components of the system are the mobile robot and the active vision system. These two basic units are interconnected by a computer designated *Master Processing Unit.* This unit controls the movements of the active vision system, communicates with the robot's onboard computer and is interconnected with two other computers designated *Slave Processing Units.* These units are responsible for processing the images provided by the active vision system. The connections between different processing units are represented in the diagram shown in figure 3 and a photograph of the system is given in figure 9.

The management and the interface with the system is done by a computer, connected to the *Master Processing Unit* using the serial link, with wireless modem.

3.2 System geometry

There are six referentials associated with the system's mechanical structure. For simplicity purposes the mechanical structure is divided into four groups: *BODY*, *NECK*, *CYCLOP* and *EYES.* The last three groups contain the referentials associated with the part of the structure referred to as head.

The upper part of the head structure is formed by the $EYES$ and the $CYCLOP$ groups. The $EYES$ group contains two referentials designated $\{\mathbf{E}_r\}$ and $\{\mathbf{E}_l\}$.

Located in the lower part of the head structure are the two referentials of the *NECK* group: $\{\mathbf{N}_y\}$ and $\{\mathbf{N}_z\}$. Each can be rotated around a different axis (the index of each name indicates the rotation axis), allowing the upper part of the structure to be moved with two degrees of freedom (head tilt and neck pan).

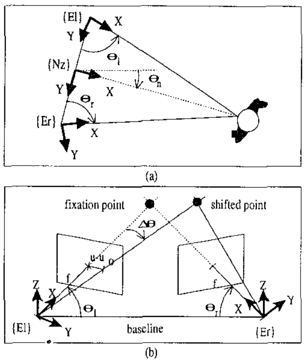

Figure 4: Cameras' vergence angle control.

Finally the *BODY* group, that contains the referential **{B}**, and is located in the mobile platform at the middle of the driving axle (i.e. the rear axle).

The head structure is placed on the mobile robot in a way that the $xz-planes$ of the referentials in the *CYCLOP*, *NECK* and *BODY* groups are coincident.

3.3 Important geometric relationships

Every time the *target* changes its position in space, it also changes its position in the images. The goal is to control the system in a way that will bring the object to the center of both images, maintaining at the same time the distance to the object. The control can be made by controlling the robot position, the neck orientation and the vergence of the two cameras. The control implies the use of these degrees of freedom, and they can be controlled simultaneously or using only one or two of them. However, it is possible to obtain expressions based in geometric relations and image perspective model, useful for this control.

The goal is to change the cameras' angles θ_l and θ_r (defined in figure 4), to maintain the projection of the *target* in the middle of the image. Since we assume that the target moves in the same plane as the mobile robot we will consider only the horizontal disparity $\Delta u = (u - u_o)$.

Let u be the coordinate in pixels of the reference point and S_x the factor between pixel and mm units. The angle that each camera must be change is given by:

$$\Delta\theta = \arctan\frac{u - u_o}{S_x f} \tag{1}$$

The position of the object to track is defined in terms of its distance D and the angle θ_n in respect to

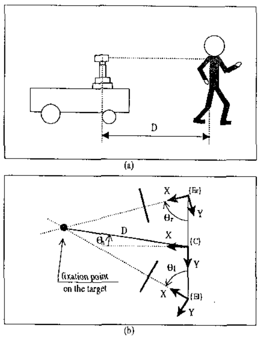

Figure 5: Distance and angle to the object defined in the plane parallel to the $xy-plane$ of the $\{\mathbf{C}\}$ referential.

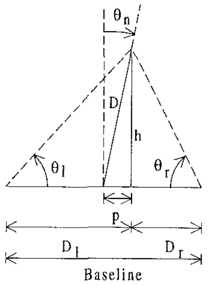

Figure 6: Auxiliar parameters.

the $\{\mathbf{C}\}$ referential, and using the *fixation* point as reference (both parameters are represented in figure 5).

To obtain the equations that give the values of D and θ_n, we start by defining the following relations, taken directly from figure 6 (equivalent to figure 5, with some auxiliary parameters).

The distance D and the angle θ_n of the *fixation* point in respect to the $\{\mathbf{C}\}$ referential can be obtained by the following equations (recall that the angle θ_n is positive clockwise - see figure 6):

$$\begin{aligned} \theta_n &= 90^\circ - \arctan\left(\frac{h}{p}\right) \\ D &= \sqrt{h^2 + p^2} \end{aligned} \tag{2}$$

Note that, when θ_l equals θ_r, the above relations are not valid. In that case, the angle θ_n is zero, and the distance D is equal to:

$$D = \frac{B}{2}\tan(\theta_l)$$

Has described above, the motion and feature detection algorithms generate the position in both images of the object to follow. Of that position, only the value along the $x - axis$ will be used, since we assume that the object *moves in the horizontal plane*, and therefore *without significant vertical shifts*.

The trajectories of the moving platform are planned by the *Master Processing Unit* based on the values of D and θ_n given by equations (2). These values D and θ_n define a 2D point with coordinates x and y in the horizontal plane where the robot moves.

4 Visual information and system state estimation

4.1 Image processing

The *Slave Processing Units* analyze independently the images captured by the frame grabbers connected to the cameras. Therefore, the motion and feature detection algorithms described are intended to work with the sequence of images obtained by each camera.

4.1.1 Gaussian pyramid

In order to speedup the computing process, the algorithms are based on the construction of a Gaussian pyramid [4]. The images are captured with 512 × 512 pixels but are reduced by using this technique. The *level* 0 corresponds to 512 × 512 pixels and *level* 4 to 32 × 32. Each pixel in one level is obtained by applying a mask to the group of pixels of the image directly bellow it. The applied mask is basically a low pass filter, that helps in reducing the noise and smoothing the images [9].

4.1.2 Image processing for saccade

The *saccade* is preceded by searching a large movement in the images. As described above, the searching phase is concerned with the detection of any types of movements, within certain limits. For that, two consecutive images (T) and ($T + 1$) separated by a few milliseconds are captured. These images are analyzed at the pyramid level 4. The analysis consists of the two steps:

- Computation of the area of motion using the images acquired at time (T) and ($T+1$). This calculus measures the ammount of shift that occurred from one frame to the other, and is used to decide when to climb or to descend levels in the pyramid.
- Absolute value subtraction of both images, pixel by pixel, generating an image of differences, and followed by the computation of the projections of the image in the x and $y - axis$. Since we assume that the target will have a small vertical motion component, only the image projection on the horizontal axis is considered for the *saccade* movement. If the movement is sensed by both cameras and its area is greater than a threshold, a *saccade* movement will be generated.

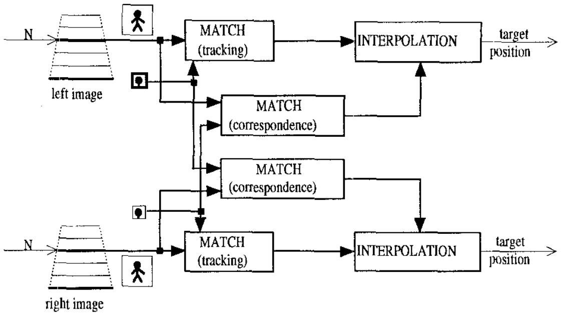

Figure 7: Illustration of the image processing used for *Fixation* and *Gaze Holding*. The visual processing consists of two phases: tracking and correspondence phase. The tracking phase locates the *target* image and gives the errors necessary to control the system and keep the object centered in the image. The correspondent phase confirms the results of the searching phase and gives the estimation of the image disparity. This disparity allows the evaluation of the distance to the object.

4.1.3 Image processing for *fixation* and *gaze holding*

The goal of *fixation* is to keep the visual *target's* image steady and centered. This presumes that the *target* is the same for the two cameras. Vergence is highly dependent of this assumption and, in this work, it is assumed that the vergence is driven by the position of the three-dimensional *fixation* point. This point corresponds to the three-dimensional position of the *target* that must be followed. This is equivalent to the problem of finding the correspondence between *target's* zones in the two images. In this work the process is called correspondence phase. Since the system is continuously controlled to keep the images centered on the fixated *target*, the correspondence zone is defined around the image center and the search process becomes easy. The correspondence used for *fixation* starts by receiving the pattern from the other *Slave Unit*. The pattern that is needed to follow in one image (left/right) is passed to the other (right/left) to find the position of the correspondent pattern. The search starts around the image center and tries to find an image that matches the pattern received.

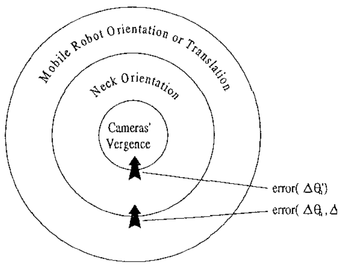

Figure 8: Graphic scheme of the principle used for the control of the system.

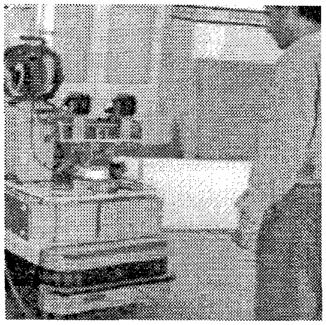

Figure 9: Obtaining the experimental results.

The *gaze holding* uses similar image processing as used in *fixation*. The *gaze holding* is implemented through *smooth pursuit* and *vergence* movements of the active vision system. These movements are based on the tracking of a pattern in each image.

4.2 Estimation of the system state

The system state can be described by two variables: the distance D from the system to the *target* and the angle θ_n of the neck pan and defined when the vision system is fixated on the *target*. Both variables have values that can be obtained by the geometry of the system and are described by equations (2).

These two quantities are continuously estimated by using a fixed coefficient filter (α-β-γ) [2]. The values estimated by this filter are described by:

$$\hat{\mathbf{x}}_D = \begin{bmatrix} D \\ \dot{D} \\ \ddot{D} \end{bmatrix} \quad and \quad \hat{\mathbf{x}}_\theta = \begin{bmatrix} \theta \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} \tag{3}$$

This filter can also be used to predict future *target* kinematics quantities. This enables the compensation for delays in the system.

5 System control

5.1 Introduction

The implementation proposed in this work is based in control loops working in parallel and based in the visual *pursuit* process. Since the inertia of the neck and the mobile robot are greater than the vergence mechanism inertia, this type of control simulates a control system with different levels - see figure 8. The inmost level comprises cameras and the vergence motors of the active vision system, responsible for tracking the *target* in real-time. At the intermediate level there is the neck sub-system, that provides the control of the orientation of the vision system and compensates the cameras' vergence movements. At the outmost level is the mobile robot sub-system that provides the compensation for the orientation of the active vision system and also controls the orientation and distance to the *target*.

Conceptually, the error between the actual distance and orientation of the *target* and the system is propagated from the outmost level to the inmost level. This gives the effect of compensation for the *target's* movements, simulating its *pursuit*.

6 Experimental results

This section gives examples of the system performances. The first example is related with the evolution of the searching algorithm. Figure 10 represents 5 cycles of the process with the entire system stopped. From top to bottom we can see the image taken at time T, the image taken at time $T+1$, the differences image and its projection on the $x - axis$. The analysis of the sequence allows us to conclude that the threshold efficiently removes the differences obtained due to the noise. It also shows that the values above the threshold, although unstable, appear in the area where the movement occurred.

The final example (figure 11) shows the importance of filtering the distance D and the angle θ. The (α-β-γ) filtering is used mostly to predict the position of the *target* one cycle in advance. This will compensate the delay between the acquisition of the images and the reaction of the system (due to processing).

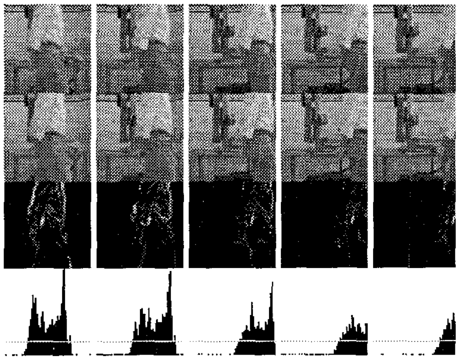

Figure 10: Evolution of the searching algorithm when the target is moving. The square in the top and bottom images are the regions in the target that system tries to pursuit.

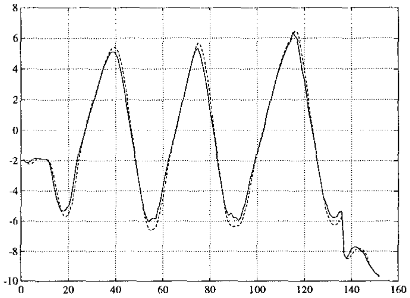

Figure 11: The angle θ. The curves have the following meaning:

- *Continuous line* - The angle measured by the system.
- *Dashed line* - (α-β-γ) filtering of the angle.

7 Summary and Conclusions

This article reports the integration of an active vision system in a mobile platform. It describes a control scheme used for real-time *pursuit* of objects moving in front of the vehicle. The *pursuit* process controls the system to maintain the initial orientation and distance to the object. The work shows that is possible to implement the system using multiple independent processes, controlling different degrees of freedom of the vision system and the mobile robot's position and orientation. A more detailed report can be retrieved from the WWW server at *http://www.dee.uc.pt/COIMBRA/artigos/index.html.*

References

[1] P. Allen, A. Timcenko, B. Yoshimi and P. Michelman, "Automated tracking and grasping of a moving object with a robotic hand-eye system," *IEEE Trans. on Robotics and Automation,* vol.9, No. 2, pp. 152-165, April 1993.

[2] S. Blackman, *Multiple-Target Tracking with Radar Applications*, Artech House, Inc, ISBN 0-89006-176-3, pp. 19-25, 1986.

[3] C. Brown, "Gaze controls with interaction and delays", *IEEE Transactions on Systems, Man and Cybernetics,* IEEE TSMC 20, pp. 518-527, May, 1990.

[4] P. Burt, C. Anderson, J. Sinniger and G. van der Wal, "A Pipelined Pyramid Machine," *Pyramid Systems for Computer Vision,* NATO ASI Series, vol. F 25, pp. 133-152, 1986.

[5] P. Burt, J. Bergen, R. Hingorani, R. Kolczynski, W. Lee, A. Leung, J. Lubin, H. Shvaytser, "Object tracking with a moving camera", *Proc. IEEE Workshop Visual Motion,* Irvine, 1989.

[6] H. Carpenter, *Movements of the Eyes,* London, Pion Limited, ISBN 0 85086 109 8, 2nd edition.

[7] D. Coombs, *Real-Time Gaze Holding in Binocular Robot Vision,* PhD Dissertation, University of Rochester, USA, June 1992.

[8] E. Grosso and D. Ballard, "Head-Centered Orientation Strategies in Animate Vision", *Tec. Report 442,* Comp. Science Dept., University of Rochester, USA, October 1992.

[9] P. Meer, E. Baugher and A. Rosenfeld, "Frequency Domain Analysis and Synthesis of Image Pyramid Generating Kernels", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. PAMI-9, No. 4, pp. 512-522, July 1987.

[10] N. Papanikopoulos, P. Khosla and T. Kanade, "Visual tracking of a moving *target* by a camera mounted on a robot: a combination of control and vision", *IEEE Trans. on Robotics and Automation,* vol.9, No. 1, pp. 14-34, February 1993.