Chapter 12

Stereo tracking system

12.1 Introduction

The object tracking is one of the most important tasks in robots that allows to work necessarily with its environment, however an important problem is that although the object has been located in the image coordinated x and y it is not possible to recover the position from the object to the robot. To solve this problem is necessary to add to the robot another camera, but this affects considerably the object tracking task.

The depth processing of stereo images requires algorithms that demand a lot of time and resources, furthermore they are highly dependent of the image size surrendered by the cameras. On the other hand to guarantee the tracking of some object, it is necessary that the controller which moves the motors of the vision system acts in intervals smaller to the time constant of the movements of the tracked object. Another important delay is in the segmentation algorithm, which usually achieves the segmentation analyzing the whole image.

Considering the above-mentioned is impossible to accomplish the tracking of objects in real time without considering that certain movements can not be tracked for the system.

In this thesis we develop a vision stereo system, which allows in an efficient way to accomplish the tracking of a soccer ball and at the same time to measure its distance. To solve this problem diverse algorithms were developed that improve the global system acting. The vision system for the stereo tracking will be identified in the successive by "stereo head system" (section 6.6).

With the objective of saving important time in the object tracking a particle filter was implemented. The particle filter allows the object identification in the image area based on the hypothesis about its position and supported for a dynamic model of the object movement, with this, the exploration to determine the object position is accomplished only in those points where it is more probable to find them and not in the whole image.

Although the system performance with the particle filter allows to save time, the camera threads make impossible the object tracking in all moment. There are occasions (particularly in speedy movements) in which it is not possible with a controller of simple structure to manipulate the motors appropriately to track to the object. In this thesis we propose two controller variants that allow to adapt the controller behavior to different conditions including to track to the object although it has disappeared of the image area due to a violent movement.

The structure of this chapter is the following: in the section 12.2 we present the stereo tracking problem, in the section 12.3 the implementation of the particle filter for the the image processing, in the section 12.4 we propose two different alternatives to control the stereo system first we analyze the controller structures that allow to adapt their behavior to different conditions determined by the object movement, later we treat the problem from the intelligent view point using fuzzy control, in the section 12.5 we analyze the problem of the lost of the tracking object in the image area and we propose the trajectory pursuit as solution, finally in the section 12.6 we present the form to calculate the object depth.

12.2 Stereo Tracking

The stereo object tracking has the same objective that the monocular object tracking [168] with the exception that in this case is also calculated the object depth. To accomplish this task is necessary to have an algorithm that allows to locate the object in the image area and a control algorithm that allows to manipulate the motors that support to the vision system in such a way that the located object stays in the image area while a stereo algorithm determines the object depth. These tasks are not different to the monocular case, only that in the stereo case the necessary time to accomplish the algorithms is bigger. The present delays in a stereo tracking system can be enumerated in the following way:

1. Stereo vision delay, this delay is due to the required time to maintain two video threads in real time.
2. Depth algorithm delay, is the delay caused by the algorithm used to determine the pixel depth using two captured images.
3. Segmentation delay, this delay is caused by the localization algorithm.

The three dealy types previously presented have mainly impact in the controller, which starting from the current tracking errors activate the motor signals to correct the system position, so that the located object continues centered in the image plane. In this thesis we present a solution to the object tracking problem with depth measurement that allows under delay conditions to have a good performance in real time. The figure 12.1 illustrate the tracking stereo problem.

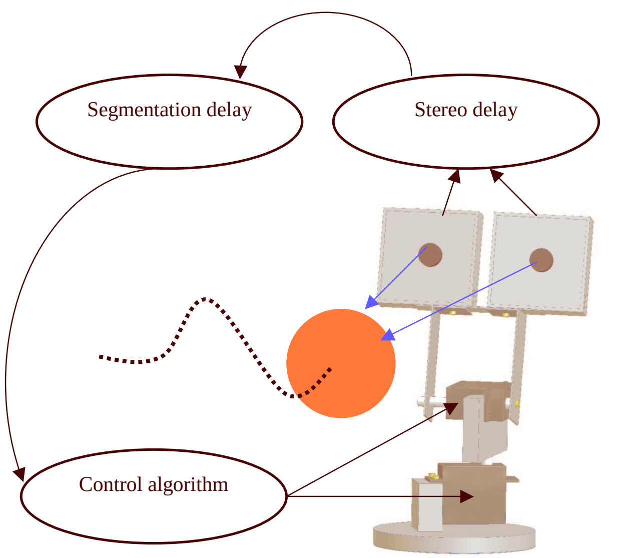


Figure 12.1: The tracking stereo problem.

12.3 Particle filter

As it was treated in the chapter 3, particle filtering [159]-[161], [167] was originally developed to track objects in clutter. The basis of the method is to construct a sample-based representation of the entire probability density function. Multiple hypothetical state (particles) of the variable of interest $\mathbf{x}_k$ (ball centroid x, y) are used, each one associated with a weight that signifies the quality of that specific particle. These sample points completely capture the true mean and covariance of the random variable and, when propagated through the nonlinear system that models the dynamics of the movement, captures the posterior mean and covariance accurately to second order (Taylor series expansion) for *any* nonlinearity. An estimate of the variable of interest is obtained by the weighted sum of all the particles. The particle filter algorithm is recursive in nature.

Color distributions are used as target models as they achieve robustness against non-rigidity, rotation and partial occlusion. We determine the color distribution inside an upright circular region centered in $\mathbf{x}^i$ with radius r=10. The color distribution $p_y = \{p_y^{(u)}\}_{u=1,2,3,\ldots m.}$at location y is calculated as

$$p_{\mathbf{y}}^{(u)} = f \sum_{i=1}^{I} k \left(\frac{\left| \mathbf{y} - \mathbf{x}^i \right|}{r} \right) \delta[h(\mathbf{x}^i) - u] \tag{12.1}$$

We need a similarity measure which is based on color distributions. A popular measure between two distributions $p(u)$ and $q(u)$ is the Bhattacharyya coefficient.

$$\rho[p,q] = \sum_{u=1}^{m} \sqrt{p^{(u)} q^{(u)}} \tag{12.2}$$

As distance between two distributions we define the measure

$$d = \sqrt{1 - \rho[p,q]} \tag{12.3}$$

The proposed tracker employs the Bhattacharyya distance to update the a priori distribution calculated by the particle filter. Each sample of the distribution represents a circle with radius *r* and is given as

$$\mathbf{x}_k^i = [\ x_k^i \quad y_k^i \quad \Delta x_k^i \quad \Delta y_k^i\ ] \tag{12.4}$$

where *x*, *y* specify the location of the circle, Δx and Δy the motion. The sample set is propagated through the application of the dynamic model

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \\ \Delta x_{k+1} \\ \Delta y_{k+1} \end{bmatrix} = \begin{bmatrix} \exp\left(-\frac{1}{4}(x_k + 1.5\Delta x_k)\right) \\ \exp\left(-\frac{1}{4}(y_k + 1.5\Delta y_k)\right) \\ \exp\left(-\frac{1}{4}\Delta x_k\right) \\ \exp\left(-\frac{1}{4}\Delta y_k\right) \end{bmatrix} + \mathbf{w}_k \tag{12.5}$$

where $\mathbf{w}_k^i$ is a multivariate Gaussian random variable.

As we want to favor samples whose color distributions are similar to the target model, small Bhattacharyya distances correspond to large weights

$$b^i = \frac{1}{\sqrt{2\pi\sigma}} \exp\left(-\frac{(1 - \rho[p_{S_k^i}, q])}{2\sigma}\right) \tag{12.6}$$

that are specified by a Gaussian with variance σ=2.5. During filtering, samples with a high weight may be chosen several times, leading to identical copies, while others with relatively low weights may not be chosen at all.

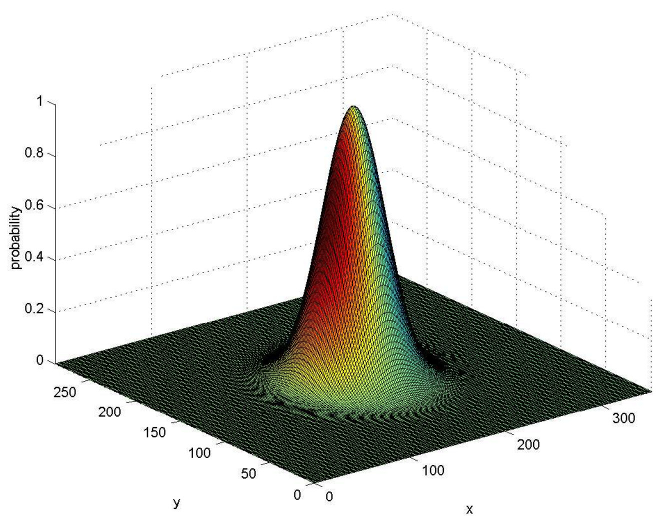
Given a particle distribution S_k^i, we need to find the state which defines with accuracy the object position. We use for the ball tracking the best particle (the $\mathbf{x}_k^j$ such that $b_k^j = \max(b_k^i) : i = 1, 2, ...M$). The figure 12.2 shows the samples distribution around the ball, the target histogram *q* and the produced probability distribution .

12.4 System control

Conventional PID controllers are only efficient where the system to be controlled is characterized by constant parameters applicable at all operating points. They are not recommended for systems operating in variable environments and/or featuring variable parameters. Evidently control of tracking systems is part of these systems, as consequence we look for a control system which

(a)

(b)



(c)

Figure 12.2: (a) The samples distribution around the ball, (b) the target histogram q and (c) the produced probability distribution .

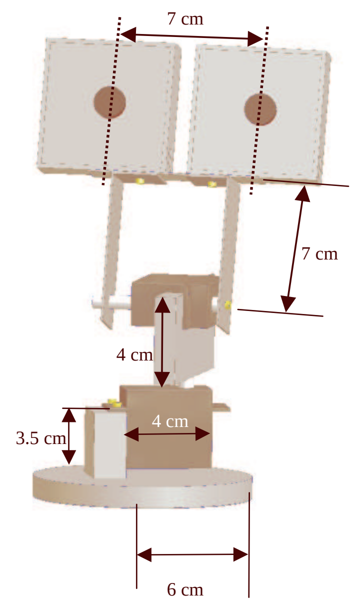


Figure 12.3: Stereo head system

ensures quasi-optimal system performance in the presence of a model with time-varying parameters.

As it is treated in the section 6.6, the stereo head system (figure 12.3) consists of two aluminium links coupled to two motors in such a way that the complete system has two degrees of freedom. With this configuration controlling appropiately the movement of the motor 1 can be accomplished the object tracking in the x coordinate of the image, while if it is controlled appropiately the motor 2 can be accomplished the object tracking in the y coordinate of the image.

For this thesis we solve the stereo tracking problem of a soccer ball. The objective therefore is to maintain the visual contact of the object in the image frame, to achieve it, we process the surrendered image by the camera 1 (the right camera of the system stereo), then we calculate the *error* in the coordinated x and y, considered as the difference between the current position of the ball and the central point of the image. According to the problem characteristics we can divide the problem in two independent controllers, one for each coordinated axis.

We present two approaches to solve the control problem of the stereo robotic head, the first approach is based on the application of a adaptive structure while the second consider a fuzzy controller.

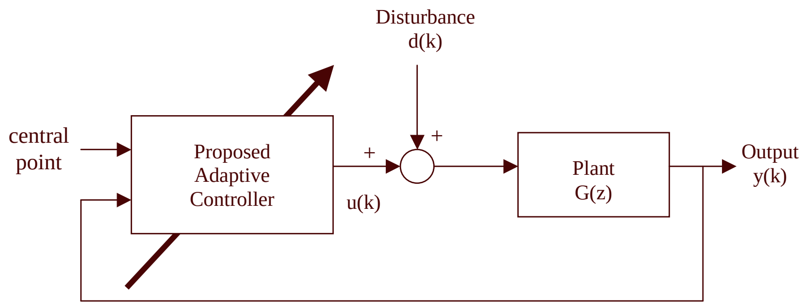


Figure 12.4: Representation of the system plant-controller.

12.4.1 Adaptive Control

Adaptive control (Chapter 6) is a set of techniques for the automatic, on-line, real-time adjustment of control-loop regulators designed to attain or maintain a given level of system performance where the controlled process parameters are unknown and/or time-varying [165]. Adaptive control is based entirely on the following hypothesis: the process to be controlled can be parametrically modelled. The adaptive control algorithm is then designed in accordance with the structure of the system model. This control algorithm can be seen as a combination of two algorithms. An *identification algorithm* uses measurements made on the system and generates information (a succession of estimates) for input to a *control law* computation algorithm. This second algorithm determines, at each instant, the control signal to be applied to the system.

The basic principle underlying adaptive control systems is relatively simple. An adaptive control system measures a certain performance rating of the system to be controlled. Starting with the difference between the desired and measured performance ratings, the adjustment system modifies the parameters of the adaptive controller in order to maintain the system performance rating close to the desired value(s).

An adaptive controller may be of conventional design or it may be more complex in structure, including adjustable coefficients such that their tuning, using a suitable algorithm, either optimizes or extends the operating range of the process to be regulated. The different methods of adaptive control differ as to the method chosen to adjust (or tune) the control coefficients.

We consider each control problem (one for each axis x and y) as a regulation problem (r(k)=central point) where the adaptive controller tries to eliminate the error caused by the object movement that represents the disturbance d(k) in this case. For it the adaptive controller generates a control signal that is applied directly to the system motor-camera identified by G(z). The figure 12.4 shows an illustration of the system plant-controller.

Each plant motor-camera is parametrically modeled as

$$G(z) = \frac{z^{-d}(b_0 + ... + b_m z^{-m})}{1 + a_1 z^{-1} + ... + a_n z^{-n}} = \frac{z^{-d} B(z^{-1})}{A(z^{-1})} \tag{12.7}$$

where
q^{-1} ,is the time delay operator.
$A(q^{-1}) = 1 + a_1 q^{-1} + ... + a_n q^{-1}$.
$B(q^{-1}) = b_0 + ... + b_m q^{-1}$.
d represent the time delay.
We model both systems supposing *n*=3 and *d*=1.
As it was seen in the section 6.4.4, we can consider the controller's structure as

$$U(k) = \frac{1}{b_0}\left[B_p(q^{-1})R(k) - \hat{R}(k, q^{-1})Y(k) - \underset{s}{\hat{B}}(k, q^{-1})U(k-1)\right] \tag{12.8}$$

where $B_p(q^{-1})R(k)$ represents the controller's reference, that represents in our case the image central point (CP). Considering the parametric model we would have

$$U(k) = \frac{1}{\hat{b}_0}\left[CP - \hat{r}_1 Y(k-1) - \hat{b}_{s1}U(k-1) - \hat{b}_{s2}U(k-2) - \hat{b}_{s3}U(k-3)\right] \tag{12.9}$$

where $\hat{b}_0, \hat{r}_1, \hat{b}_{s1}, \hat{b}_{s2}, \hat{b}_{s3}$, are the parameters to adapt considering as minimization approach the error $Y(k) - CP$.
Considering as vectors

$$\hat{\theta}(k) = \begin{bmatrix} \hat{b}_0 & \hat{b}_{s1} & \hat{b}_{s2} & \hat{b}_{s3} & \hat{r}_1 \end{bmatrix} \tag{12.10}$$

$$\psi(k) = \begin{bmatrix} U(k) & U(k-1) & U(k-2) & U(k-3) & Y(k-1) \end{bmatrix} \tag{12.11}$$

we apply the recursive algorithm 5 (Appendix A) that calculates in each instant *k* the controller output and the estimate parameters of the controller for the varying plant conditions.

The figure 12.5 shows the adaptive controller performance for the ball tracking considering only the *x* axis (motor 1).

12.4.2 Fuzzy controller.

A fuzzy control system (Chapter 7) is a system which emulates a human expert [166]. In this situation, the knowledge of the human operator would be put in the form of a set of fuzzy linguistic rules. These rules would produce an approximate decision, just as a human would.

Algorithm 5 Recursive algorithm for the estimate of controller parameters.

1.To set the initial values for $\Gamma(k)$ and $\overset{\wedge}{\theta}(k)$.
$\Gamma(k) = 1$, this means that in the beginning the uncertainty is high.
$\overset{\wedge}{\theta}(k)$, is initialized with random values between -1 and 1.
2. To obtain the value of $\mathrm{U}(k)$.
3. To obtain the value of $\mathrm{Y}(k)$.
4. The error is calculated $\mathrm{e}(k) = \mathrm{Y}(k) - \mathrm{CP}$.
5. To calculate $F(k) = \frac{\Gamma(k)\psi(k)}{1+\psi^T(k)\Gamma(k)\psi(k)}$.
6. New parameters estimate $\overset{\wedge}{\theta}(k+1) = \overset{\wedge}{\theta}(k) + F(k)\mathrm{e}(k)$.
7. We calculate the new $\Gamma(k)$.

$$\Gamma(k+1) = [\mathbf{I} - F(k)\psi^T(k)]\Gamma(k)$$

8. We update the vector $\psi(k)$.
9. *k*=*k*+1 and we return to the step 2.

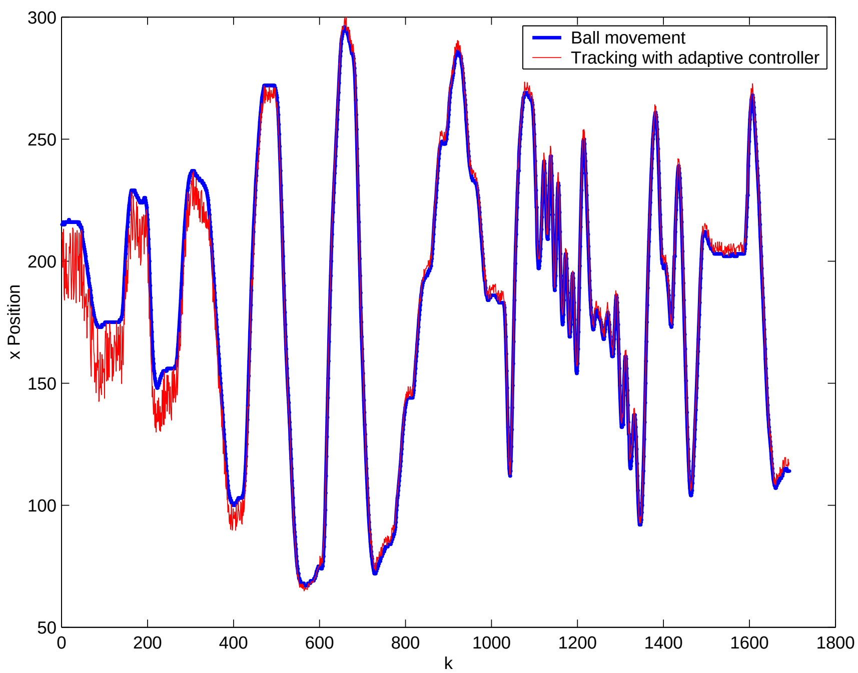


Figure 12.5: Adaptive controller performance for the ball tracking.

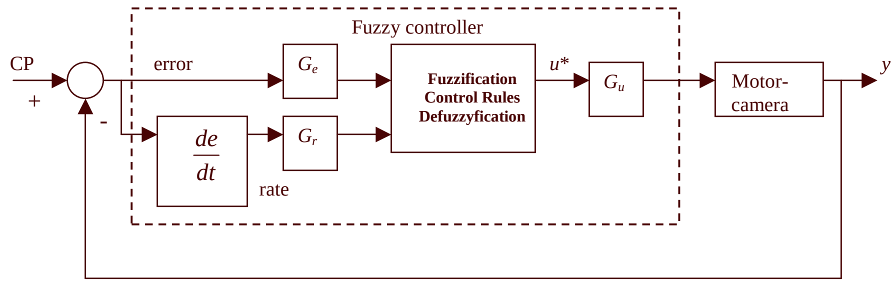


Figure 12.6: Fuzzy controller architecture.

Table 12.1: Obtained controller parameters

Axis	*Ge*	*Gr*	*Gu*
x	2.1	3	0.1
y	5	4	0.1

In the last section we treated the stereo head system control applying an adaptive control approach, now in this section we will apply a fuzzy condensed controller (section 8.2) to accomplish the same task.

The controller's architecture is illustrated in the figure 12.6, as is shown, there are two inputs to the controller: error and rate. The fuzzy controller has a single output, which is used to control the system motor-camera. The gains *Gu*, *Ge* and *Gr* are determined by tunning and they correspond respectively to the output gains, the error and error rate gains. The u* is the defuzzyficated output, that means the "crisp output". The fuzzy linguistic rules, input and output membership functions for the fuzzy controller correspond those explained in the section 8.2 for the control of the monocular head system.

The fuzzy controllers were implemented and tuned to different dynamic parameters to control each axis. The Table 12.1 shows the parameters values. The figure 12.7 shows the fuzzy controller performance for the ball tracking considering only the x axis (motor 1).

Although the adaptive controller shows a good velocity response, in steady state it has multiple oscillations that complicate the image processing. The fuzzy controller shows a better trade off between velocity and smoothness control, for this reason we finally have chosen the fuzzy controller.

12.5 Lost object Tracking

Nevertheless the visual tracking and control algorithms implemented in the previous sections, violent movements can produce "the lost" of the tracking

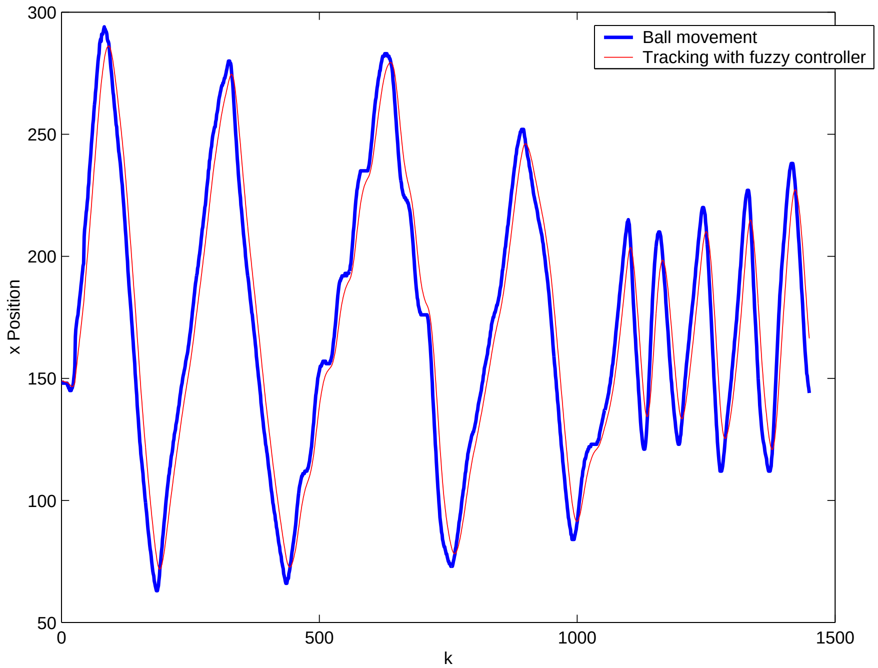


Figure 12.7: Fuzzy controller performance for the ball tracking

object. For lost we refer that the object leaves the image area in such a way that it cannot be located. To be able to accomplish the tracking in those situations we propose an algorithm which starting from the previous trajectory and the modification of the control law allows to find the object and to continue with the tracking.

During the tracking system operation the position of the 50 previous points are stored in the vector **p**. If the tracking object is lost, we will use those points to find the object direction. We calculate the direction equation using the initial and final point of the vector **p**. The point numbers in the vector **p** determines the precision with which the equation can be calculated.

If we consider $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(50)}$ the first and last elements of the vector **p**, the equations can be calculated in the following way

$$\mathbf{p}^{(1)} = \begin{bmatrix} p_x^{(1)} & p_y^{(1)} \end{bmatrix} \tag{12.12}$$

$$\mathbf{p}^{(50)} = \begin{bmatrix} p_x^{(50)} & p_y^{(50)} \end{bmatrix} \tag{12.13}$$

$$c_y = \frac{p_y^{(50)} - p_y^{(1)}}{p_x^{(50)} - p_x^{(1)}}(1.5Ca_x - p_x^{(1)}) + p_y^{(1)} \tag{12.14}$$

$$c_x = \frac{(1.5Ca_y - p_y^{(1)})(p_x^{(50)} - p_x^{(1)})}{p_y^{(50)} - p_y^{(1)}} + p_x^{(1)} \tag{12.15}$$

where c_x and c_y represent the points of the direction trajectory. With these equations we can extrapolate and to find a point *outside* of the image area that can be considered as "the object position".

If we consider Ca_x and Ca_y the dimensions of the image area , we would have three different extrapolation cases.

Case 1.

When $\left|p_x^{(50)} - p_x^{(1)}\right| < 30$, the extrapolated point is $Ep = \begin{bmatrix} p_x^{(50)} & 1.5Ca_y \end{bmatrix}$.

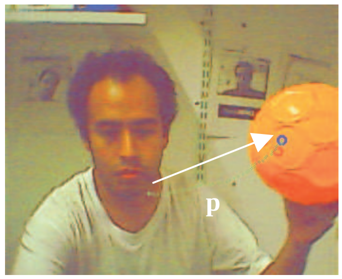
Case 2.

When $\left|p_y^{(50)} - p_y^{(1)}\right| < 30$, the extrapolated point is $Ep = \begin{bmatrix} 1.5Ca_x & p_y^{(50)} \end{bmatrix}$.

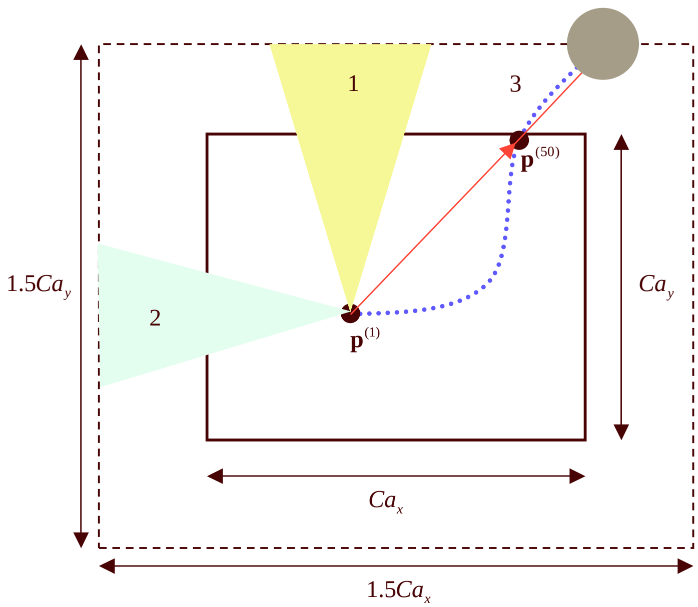
Case 3.

Otherwise condition, in this case we apply the equations 12.13 and 12.14 in such a way that $Ep = \begin{bmatrix} c_x & c_y \end{bmatrix}$.

The figure 12.8(a) shows the points storage during the tracking operation and the figure 12.8(b) a representation of the proposed extrapolation.



(a)



(b)

Figure 12.8: (a) The points storage during the tracking operation and (b) representation of the proposed extrapolation.

The extrapolated point *Ep* will be used in the error calculation until the object is found by image processing. Considering the above-mentioned the input of the fuzzy controllers will be

For motor 1

$$error_{motor1} = \mathrm{CP} - Ep_x \tag{12.16}$$

For motor 2

$$error_{motor2} = \mathrm{CP} - Ep_y \tag{12.17}$$

Applying these errors, the controllers outputs will be so high that having even found the object, the dynamic inertia will destabilize the controller. To avoid this problem we propose the application of a gain T that multiplies to the output controller, which increases from 0.01 to 1 depending on the sample instant k. Obviously the gain T will be configured automatically at 1 when the object has been located. The figure 12.9 summarize the proposed algorithm for the lost object Tracking.

12.6 The tracking object distance

Calculating the distance (Chapter 11) of various points in the scene relative to the position of the camera is one of the important tasks for a computer vision system. A common method for extracting such depth information from intensity images is to acquire a pair of images using two cameras displaced from each other by a known distance.

The geometry of binocular stereo is shown in Figure 12.10. The model has two identical cameras separated only in the X direction by a *baseline distance b*. The image planes are coplanar in this model. A feature in the scene is viewed by the two cameras at different positions in the image plane. The displacement between the locations of the two features in the image plane is called the *disparity*. The plane passing through the camera centers and the feature point in the scene is called the *epipolar plane*. The intersection of the epipolar plane with the image plane defines the *epipolar line*. For the model shown in the figure, every feature in one image will lie on the same row in the second image. In practice, there may be a vertical disparity due to misregistration of the epipolar lines. Many formulations of binocular stereo algorithms assume zero vertical disparity.

A necessary intermediate step for the distance determination of an object is the camera calibration of the stereo system. The camera calibration problem (Chapter 10) is to relate the locations of pixels in the image array to points in the scene. Since each pixel is imaged through perspective projection, it corresponds to a ray of points in the scene. The camera calibration problem is to determine the equation for this ray in the absolute coordinate system of the scene. The camera calibration problem includes both the exterior and interior orientation problems. The focal distance **f** is a important parameter used for

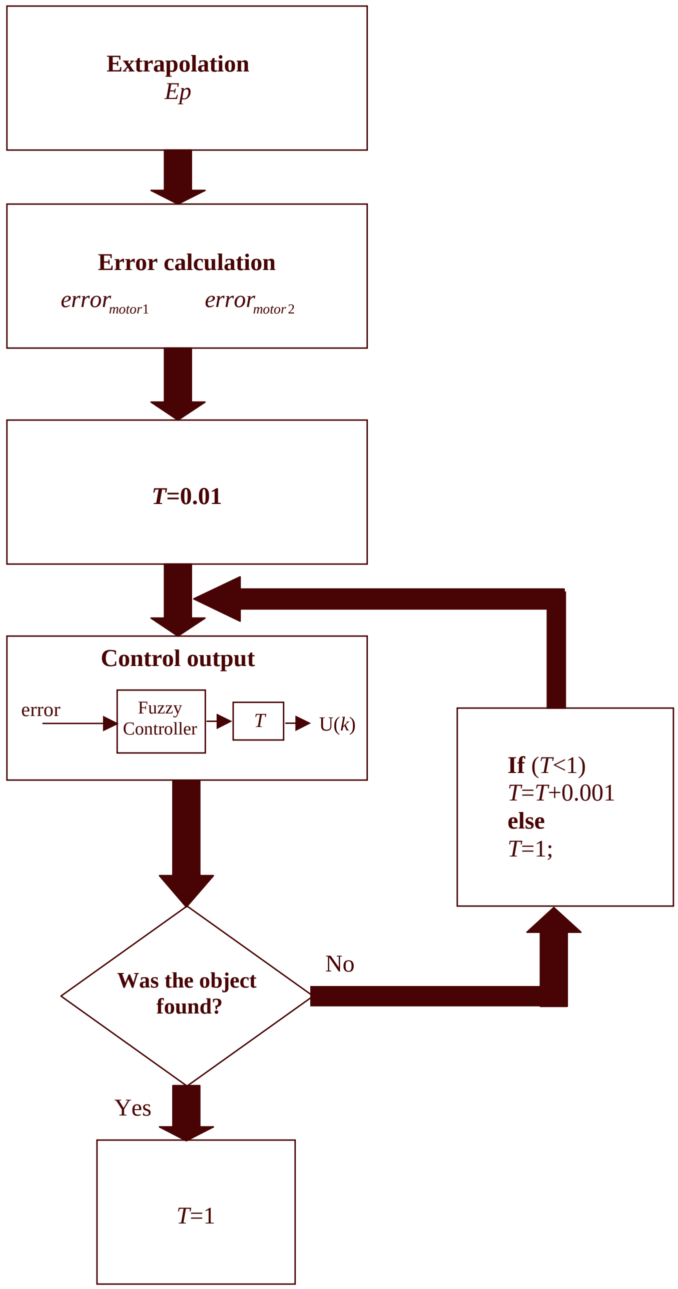


Figure 12.9: Algorithm for the lost object Tracking

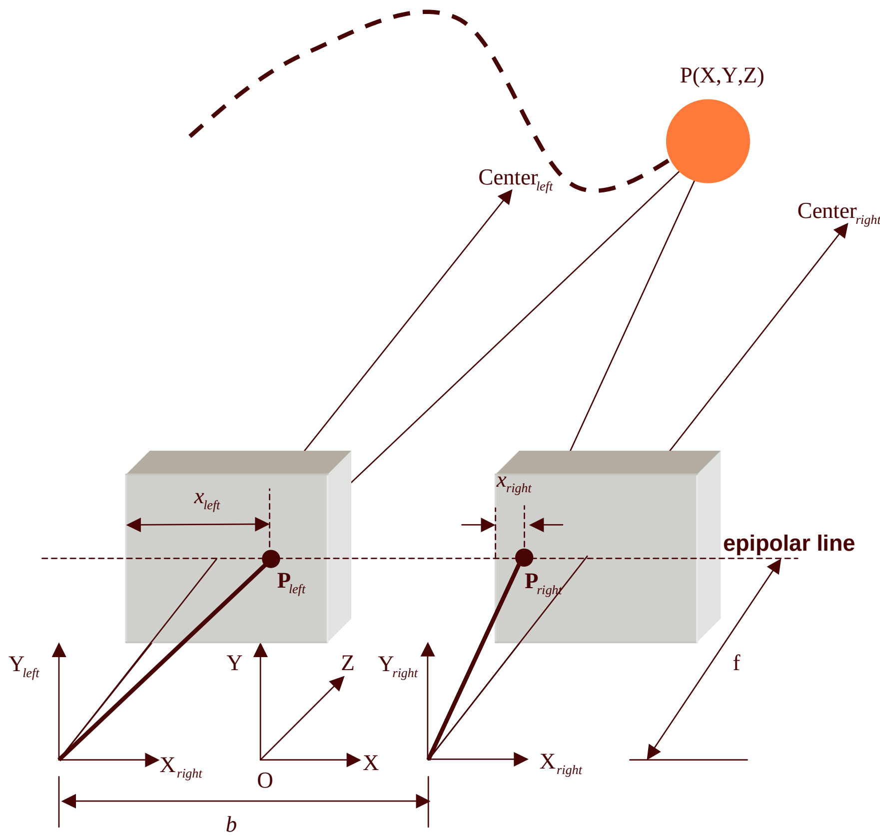


Figure 12.10: Geometry stereo model

Algorithm 6 Depth determination of the object tracking.

1. We determine the object position with the vision algorithm (Particle filter) for each camera. The position for the left camera is defined as $\mathbf{P}_{left}$ and for the right $\mathbf{P}_{right}$.
2. We apply the control law for both motors.
3. If $error_{motor1} < 5$ **then**

$$\mathrm{Z} = \frac{\mathrm{f}b}{x_{left} - x_{right}}$$

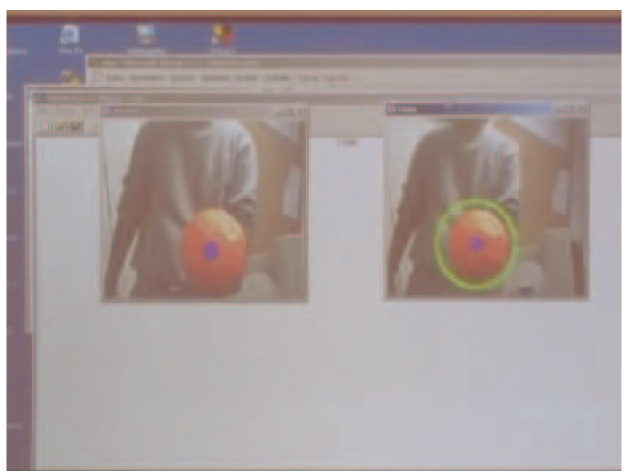
else
The distance cannot be calculated
4. To return to the step 1.

the distance determination that is calculated as intrinsic parameter by the calibration method.

For the depth determination of the tracked object , we accomplish the procedure represented in the algorithm 6. The object depth can be determined only when the error in the control algorithm of the motor 1 has a small value, it is also evident that when the object is not found by the vision algorithm, it is not possible to find the distance.

All the algorithms were programmed in C++ and proven in a PC 900 MHz with 128 MB in RAM. The figure 12.11(a) shows an image of the stereo robotic head working, while the figure 12.11(b) shows a software capture.

(a)



(b)

Figure 12.11: (a) Stereo robotic head working and (b) one software capture