Chapter 8

Gaze Control

8.1 Introduction

In the robotics area, visual tracking is an important and difficult problem therefore is necessary to have a robust and efficient control algorithm which presents immunity characteristics to stochastic direction and speed changes of the object to be tracked. Also is important count with a segmentation algorithm which be able to tolerate changes in the intensity of light. We describe in this chapter the implementation of fuzzy controllers based on the fuzzy condensed algorithm. For this thesis we used two fuzzy condensed algorithms running in a PC to control a robot's head which tracks a human face. We describe the main lines of the fuzzy condensed algorithm as well as the LVQ neural networks architecture employed and the implementation, the fuzzy condensed controller performance in comparison to a PID controller and real time results.

Fuzzy logic (as it was explained in the chapter 5) allows partial truths and multivalue truths. It is therefore especially advantageous for problems which cannot be easily represented by mathematical modelling because data is either unavailable, incomplete, or the process is too complex. The real-world language used in fuzzy control enables the incorporation of approximate human logic into computers. Using linguistic modelling, as opposed to mathematical modelling, greatly simplifies system design and modification. It generally leads to quicker development cycles, easy programming, and fairly accurate control. However it is important to underline the fact that fuzzy logic solutions are usually not aimed at achieving the computational precision of traditional techniques, but aims at finding acceptable solutions in shorter time.

8.2 Fuzzy condensed control scheme

In this chapter the fuzzy condensed algorithm is used as described in [121](fuzzy PD) witch consists of only 4 rules and has the structure illustrated in the Fig. 8.1.

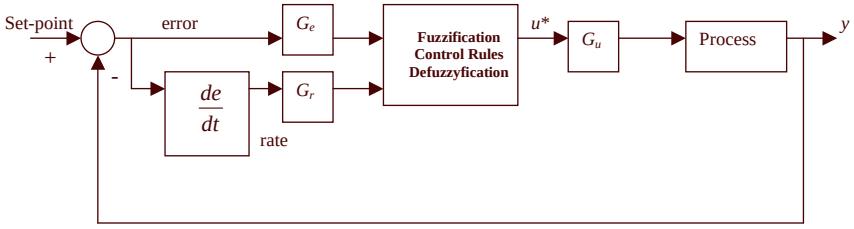


Figure 8.1: Fuzzy-condensed structure

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".

8.2.1 Fuzzyfication

As is shown in Fig. 8.1, there are two inputs to the controller: $error$ and $rate$. The error is defined as:

$$error = setpoint - y \tag{8.1}$$

Rate it is defined as it follows:

$$rate = (ce - pe)/sp \tag{8.2}$$

Where ce is the current error, pe is the previous error and sp is the sampling period. Current and previous error, are referred to an error without gain. The fuzzy controller has a single output, which is used to control the process. The input and output membership functions for the fuzzy controller are shown in Fig. 8.2 and Fig. 8.3, respectively. Fig. 8.2 shows that each input has two linguistic terms. For the error input are: $Ge * negative\ error$ (en) and $Ge * positive\ error$ (ep) and for the rate input are: $Gr * negative\ rate$ (rn) and $Gr * positive\ rate$ (rp), while the output fuzzy terms are shown in Fig. 8.3 and they are: $Negative\ output$ (on), $Zero\ output$ (oz) and $Positive\ output$ (op).

As shown in Fig. 8.2, the same function is applied for $error$ and $rate$ but with different scaling factors: Ge and Gr respectively.

In Fig. 8.2 and Fig. 8.3, H and L are two positive constants to be determined. For convenience we will take H=L to reduce the number of control parameters to be determined.

The membership functions for the input variables, $error$ and $rate$, are defined by the following expressions [3]:

$$\mu_{ep} = \frac{L + (G_e * error)}{2L}$$

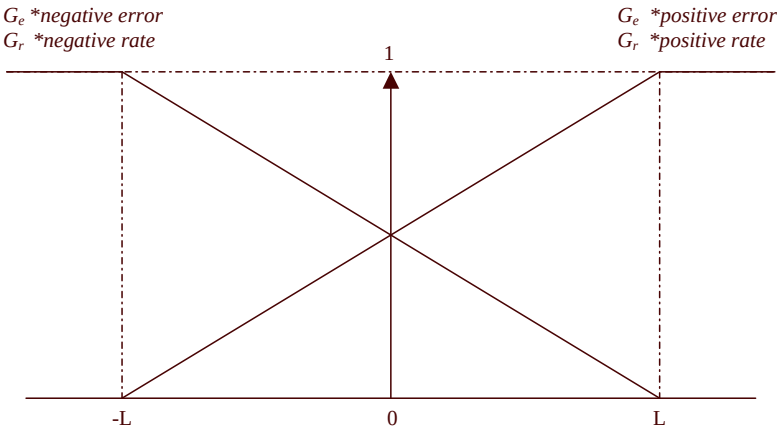


Figure 8.2: Input membership functions

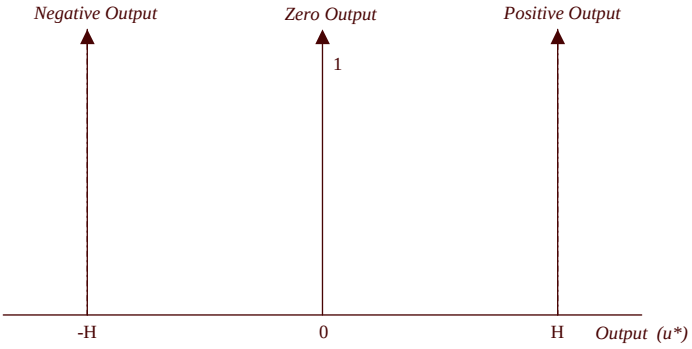


Figure 8.3: Output membership functions

$$\mu_{rp} = \frac{L + (G_r * rate)}{2L}$$

$$\mu_{en} = \frac{L - (G_e * error)}{2L}$$

$$\mu_{rn} = \frac{L - (G_r * rate)}{2L} \tag{8.3}$$

8.2.2 Fuzzy rules

Since here $error$ will correspond to $Ge * error$ and $rate$ to $Gr * rate$.

Exist four rules to evaluate the condense fuzzy controller [4]:

R1. If $error$ is ep and $rate$ is rp then $output$ is op

R2. If $error$ is ep and $rate$ is rn then $output$ is oz

R3. If $error$ is en and $rate$ is rp then $output$ is oz

R4. If $error$ is en and $rate$ is rn then $output$ is on

The determination of these rules can be accomplished easily if the system evolution is analyzed in the different operation points. For example, when the *error* and the *rate* increase (rule 1), it means that the system response decreases and moves away from the *setpoint* for this reason is necessary to apply a *positive* stimulus that allows to increase the system output. The figure 8.4 shows the determination of the different rules based on the system response.

8.2.3 Defuzzyfication

The defuzzyfication method used is the gravity center, in this case is:

$$u = \frac{-\mathrm{H}(\mu_{R4}) + 0(\mu_{R2} + \mu_{R3}) + \mathrm{H}(\mu_{R1})}{\mu_{R1} + \mu_{R2} + \mu_{R3} + \mu_{R4}} \tag{8.4}$$

For the fuzzy condensed controller proposed, the input error and rate values ranges can be represented in 20 input region (IC), like is shown in Fig. 8.5.

If the membership functions are evaluated, the 4 control rules, H=L and the defuzzyfication is applied in each one of the 20 inputs combinations, then 9 equations can be obtained [123], which can determine the control signal u that should be applied, depending on the region in which is. In other words, to implement the fuzzy condensed controller, only will be necessary to know the region in which the inputs variables are and later evaluate the corresponding equation for this region. For example the first equation acts in regions IC1, IC2, IC5, IC6. The figure 8.6 shows the control surface of the fuzzy condensed controller considering H=L=1.

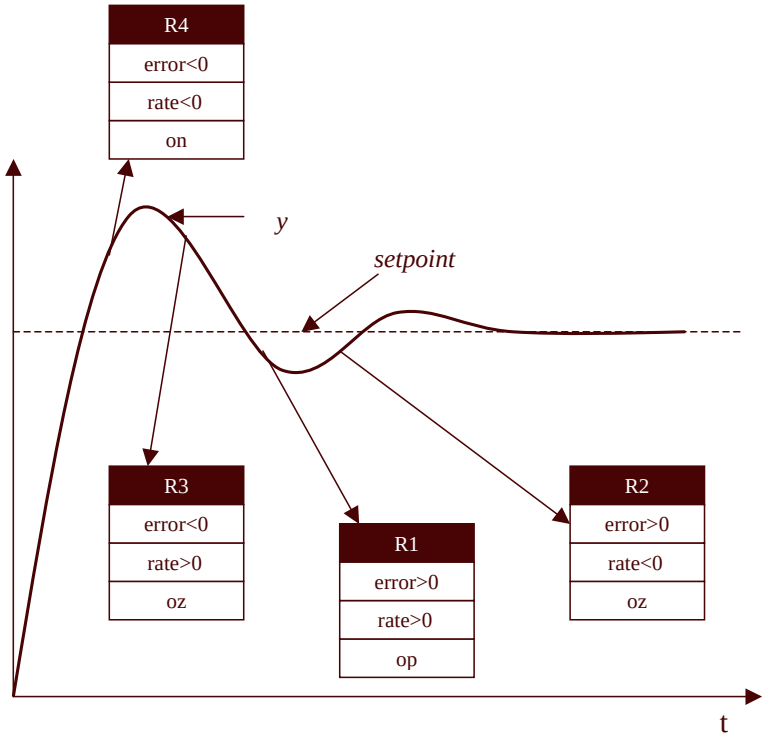


Figure 8.4: Determination of the different rules based on the system response.

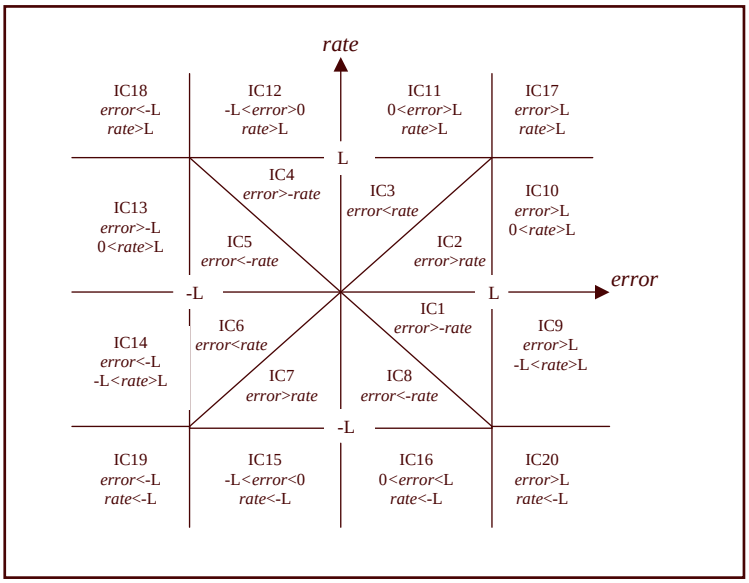


Figure 8.5: Input regions

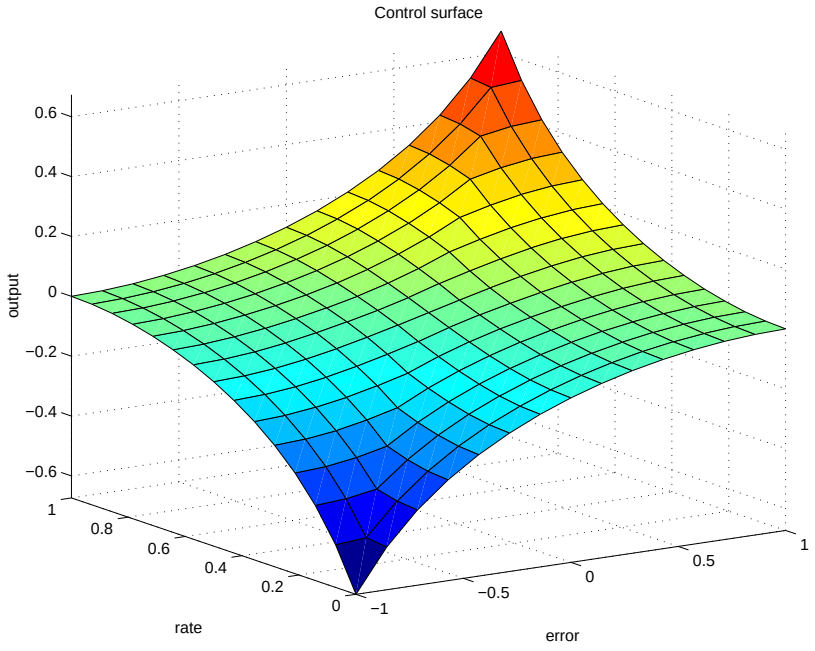


Figure 8.6: Control surface of the fuzzy condensed controller considering H=L=1.

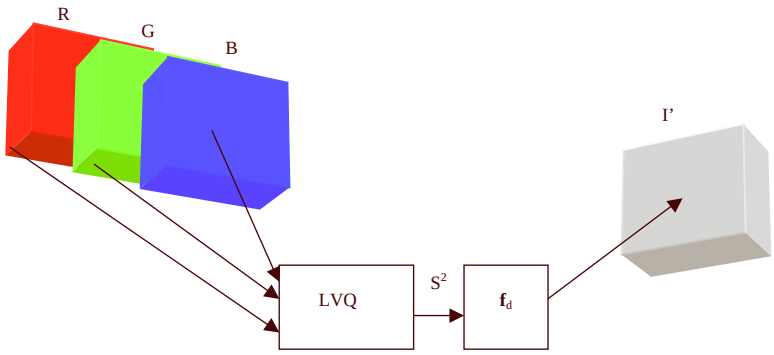


Figure 8.7: Architecture of the color segmentation system

8.3 Architecture of the color segmentation System

The core of the algorithm is a LVQ network whose inputs are connected directly with the pixel-vector of the image $\mathbf{I}$ (in RGB format) and its outputs are connected directly to the decision function $\mathbf{f_d}$, which produces an output of 1 or 0 depending of if the color corresponds to the object to be segmented forming in this way a new vector image $\mathbf{I'}$. The Fig. 8.7 shows a scheme of the segmentator.

Considering that the LVQ net is configured with N output neurons, then it would be possible train the competitive net to learn the configuration space and the color pixels topology, described as the vector $\mathbf{p}$ with elements $\mathbf{p}_R$, $\mathbf{p}_G$ and $\mathbf{p}_B$ coming from the image. For the supervised training of the linear net, we suppose that the image $\mathbf{I}$ contains an Object O to be segmented, being $\mathbf{p_O}$ a pixel corresponding to the object, we train the linear network in such a way that the class $\mathbf{S}^2$ (of this pixel) corresponds to the assigned (in a supervised way) for the neuron $N/2$ of the linear network.

The idea is that the color to be segmented is located halfway of the network, while the similar colors are located in the neighboring neurons. In such a way that if exists a vector $\mathbf{p_1}$ that belongs to the object but for the illumination conditions and noise could be classified in the neighborhood of the neuron $N/2$.

The classification achieved by the LVQ network gives a vector of elements categorized by $\mathbf{S}^2$ of N elements corresponding to the N classes. Each element of the $\mathbf{S}^2$ vector could have 2 possible values, 1 or 0 and only an element from each vector could be 1, while the other elements will be 0. Then for the object to be segmented, the activation of the neurons is concentrated in the middle of the vector, that is the neurons nearest to the $N/2$ will have a bigger possibility to be activated than the other ones corresponding to the color to be segment.

Considering the previous problem is necessary to define a function $\mathbf{f_d}$ able to define neuron´s density which will be taken to consider if a pixel corresponds or not to the color to be segmented, this function will be called in this

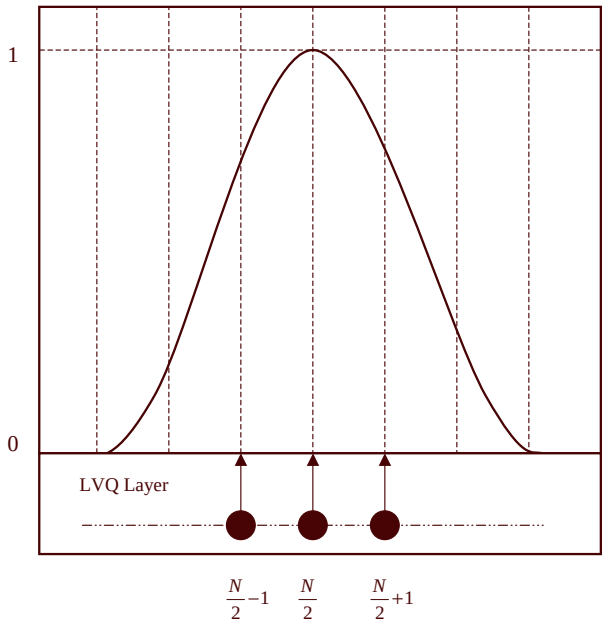


Figure 8.8: Decision function model

work decision function. It is possible to formulate many functions which could solve the decision problem satisfactorily, however the Gaussian function has been chosen by its well-known properties, although it is possible to use other, including non-simmetrical distributions. The Fig. 8.8 shows graphically the function used.

The equation 8.5 shows mathematically this function where $\mathbf{g}$ is the number of the activated neuron, μ is $\mathrm{N}/2$ and σ is the standard deviation. Therefore, $\mathbf{f_d}$has only a calibration parameter represented by σ which determines the generalization capacity of the complete system. Thus, if the value of σ is chosen small enough, the segmentation capacity will be more selective than in the case of a bigger σ. For the final result the decision function was evaluated with a threshold of 0.7.

$$\mathbf{f_d}(\mathbf{g}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mathbf{g}-\mu)^2}{2\sigma^2} \right) \tag{8.5}$$

8.4 Implementation

The goal of the robotic head system is to be in permanent contact with the person's face, to achieved that, a face localization algorithm is used [124], it calculates the face's central point, then the controller acts on the difference between

Axis	Ge	Gr	Gu	L
x	2.5	4	0.1	100
y	6	4	0.1	100

Table 8.1: Obtained controller parameters

this point and the image's central point. The Fig. 8.9a shows the localization process and Fig. 8.9b the tracking system.

A condensed controller allows real time work for each captured frame, in this case even to a 30 Hz. The system was implemented with an USB Webcam with a 352x244 resolution, therefore the central point of the image and also the fuzzy controller set point is 176 for x and 144 for y. The complete vision and control system was totally implemented in C++

The robot head (section 6.6) consists of two rc-servo motors to control each axis movement. The mathematical model for the tracking system was implemented using Simulink as shown in Fig. 8.10, where Ea represents the applied voltage and Pos the position.

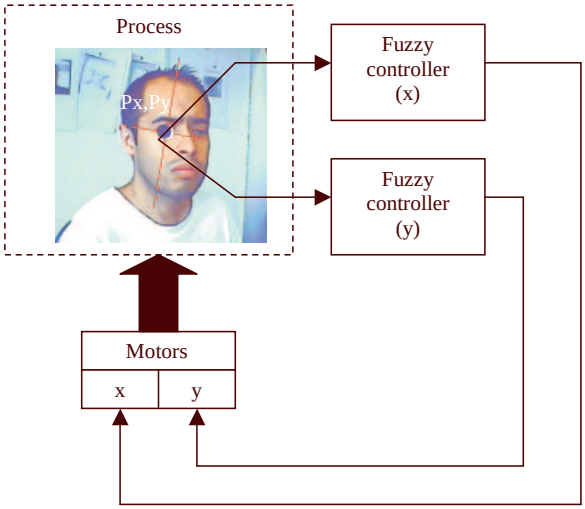
Two fuzzy controllers were implemented and tuned to different dynamic parameters to control each axis. For violent change-direction movements (when the position's rate value was considerably high) the system experimented some difficulties, which were eliminated implementing a Kalman's filter allowing the smooth position's rate. The controller's parameters were experimentally obtained, the Table 1 shows the values.

Diverse tests were realized to prove the controller's performance, these are described as follows:

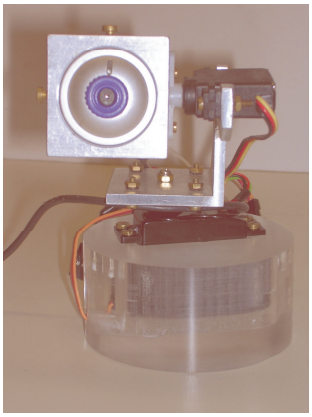
- There were applied movements in x axis (avoiding to make them in y axis), simulating a "step signal" in x direction. The Fig. 8.11(a) shows the head's position, the motor's signal and the camera's position for this case. Fig. 8.11(b) shows the head's movements map in x axis.

- There were applied movements in y axis (avoiding to make them in x axis), simulating a "step signal" in y direction. The Fig. 8.12 (a) shows the head's position, the motor's signal and the camera's position for this case. Fig. 8.12 (b) shows the head's movements map in y axis.

- There were applied violent movements in both axes to prove the system's robustness. The Fig. 8.13 (a)-(b) shows the head's position, the motor's signal and the camera's position for this case. Fig. 8.14 shows the head's movements map.



(a)



(b)

Figure 8.9: (a) Localization process. (b) Tracking System

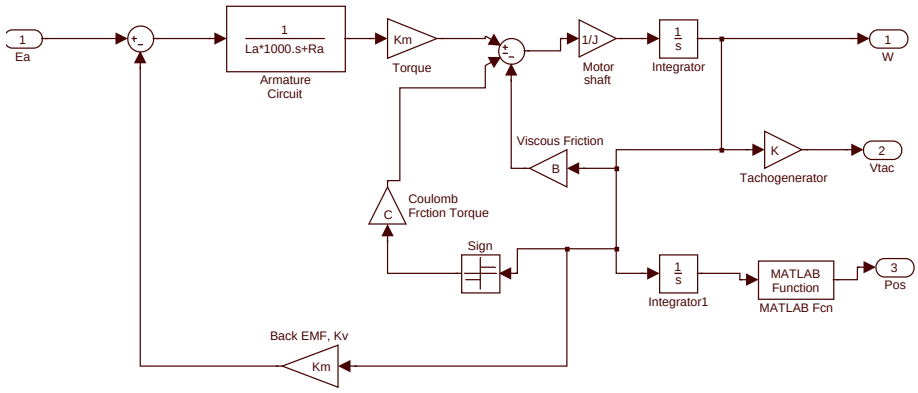
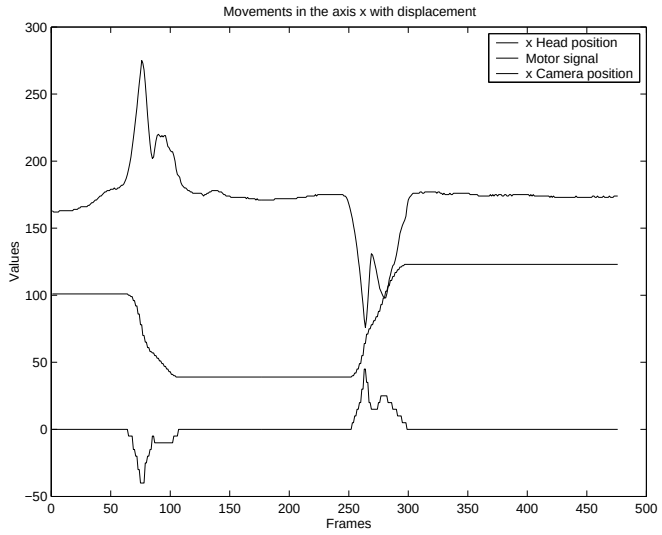


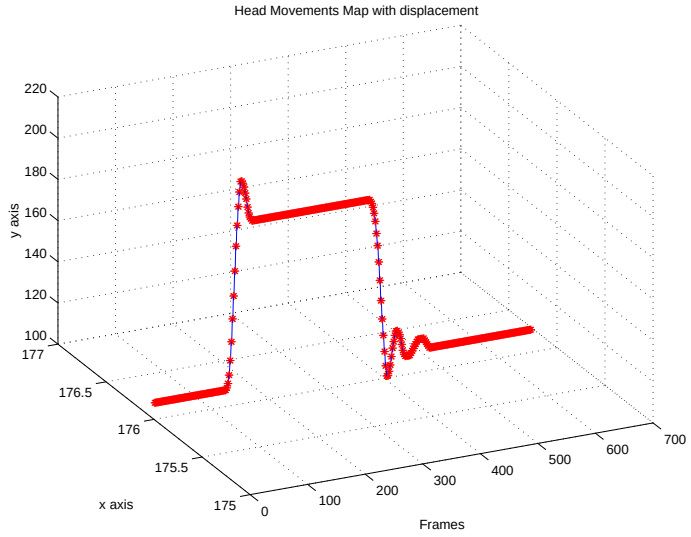
Figure 8.10: Tracking system model implemented in Simulink.

8.5 Conclusions

The fuzzy condensed controller showed a better performance compared to the PID controller. Although the PID controller behavior seems similar to the fuzzy condensed controller behavior for slow object movements, when the object to be tracked move faster, the PID controller's behavior showed a bigger settling-time, which means: the PID controller was slower to track the object as shown in the Fig. 8.15 where can also be observe that after an abrupt x axis change position the system presented extended oscillations, situation that doesn't happen with the condensed fuzzy controller as it was shown during the tests applied for abrupt movements in both axis in Fig. 8.11 to Fig. 8.12.

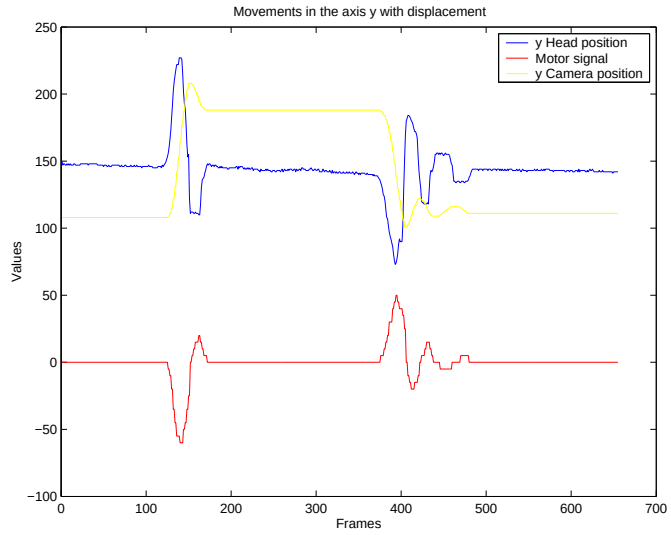


(a)

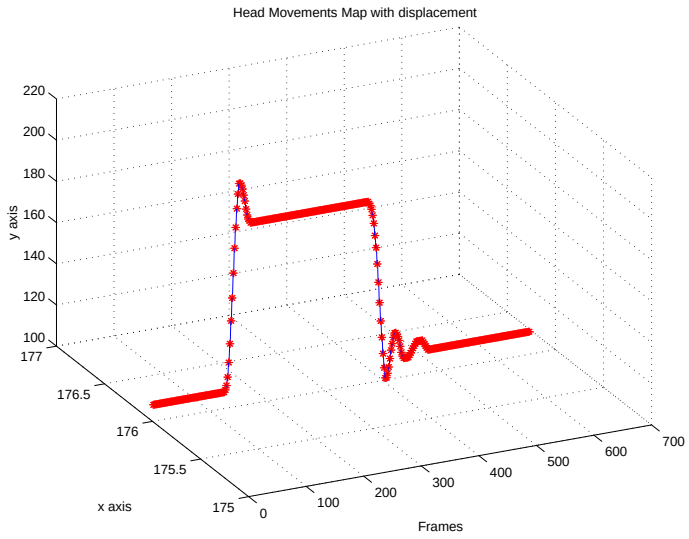


(b)

Figure 8.11: (a) Movements in x axis. (b) Movements map in x axis

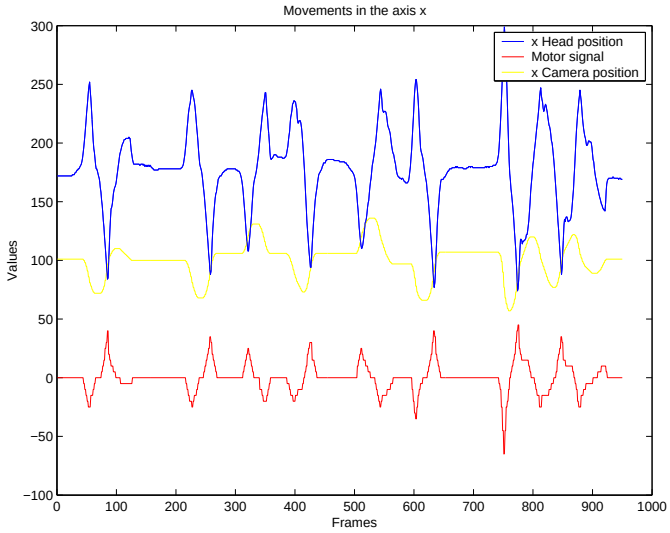


(a)

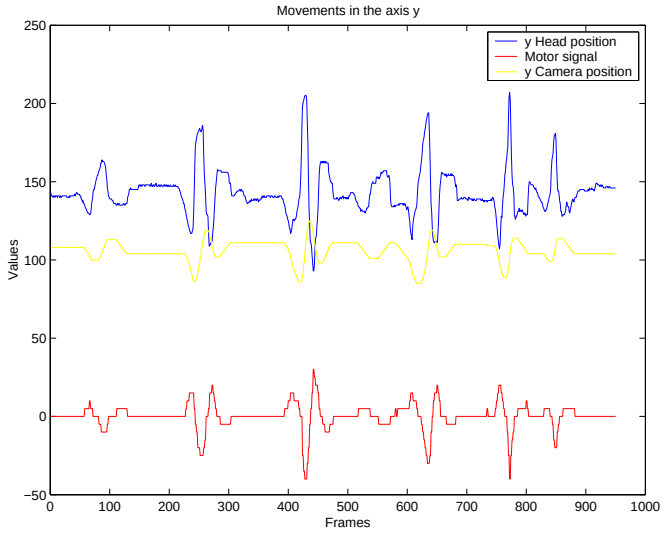


(b)

Figure 8.12: (a) Movements in y axis. (b) Movements map in y axis.



(a)



(b)

Figure 8.13: (a) Movements in x axis. (b) Movements map in y axis

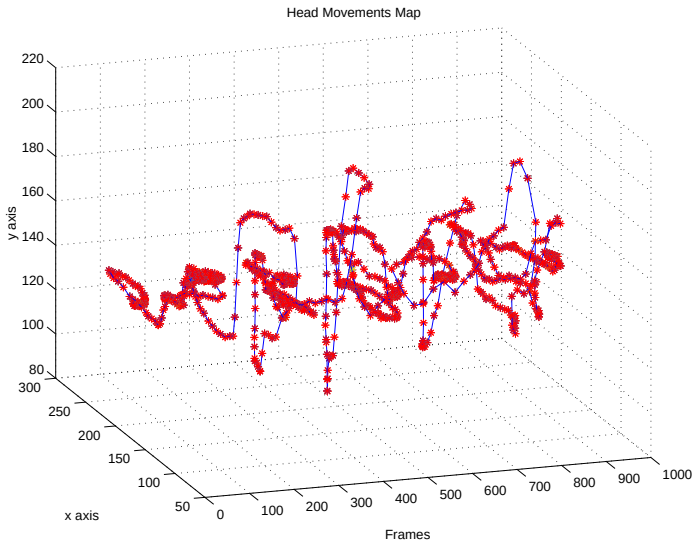


Figure 8.14: Movements map for both axis

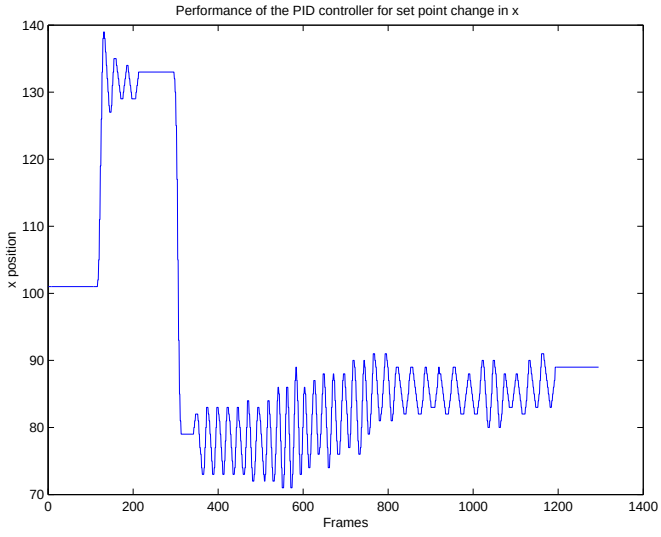


Figure 8.15: PID settling-time