Neurofuzzy prediction for visual tracking

Erik V. Cuevas[1,2], Daniel Zaldívar[1,2], Raúl Rojas [1]

[1]Institut für Informatik
Freie Universität Berlin
Takustr. 9, D-14195 Berlin, Germany
Email: cuevas, zaldivar, rojas@inf.fu-berlin.de

[2]División de Electrónica y Computación
Universidad de Guadalajara
Av. Revolución No. 1500, C.P. 44430
Guadalajara, Jal., México

Abstract

Real time visual tracking is a complicated problem due the different dynamic of the objects involved in the process. On one hand the algorithms for image processing usually consume a lot of time on the other hand the motors and mechanisms used for the camera movements are significantly slow. This work describes the use of ANFIS model to reduce the delay's effects in the control for visual tracking and also explains how we resolved this problem by predicting the target movement using a neurofuzzy approach.

1. INTRODUCTION

Real time visual tracking is a complicated problem due the different dynamic of the objects involved in the process. On one hand the algorithms for image processing usually consume a lot of time and on the other hand the motors and mechanisms used for the camera movements are significantly slow. A tracking system is composed of 3 connected systems, first it has an algorithm which receives as input the captured image by a camera, the algorithm processes the image segmenting and locating the object of interest, the localization of the object can be considered as the output of this block. The following block is the controller which takes as input the object localization. The controller tries to maintain the object into the visual frame, therefore it sends the appropriate signals to the mechanisms which can manipulate directly the position. The Fig. 1 shows a visual tracking system representation. Therefore, the system can be considered as a feedback system control where the elements which participate have different dynamic characteristics.

The tracking systems developing supposes a challenge in the controller's design. This, should possess the capacity to be robust and immune to noises due the object movement and besides to be able to work with a inherent delay. The delay is in fact the delays sum produced in two system blocks. The image capture and the image processing are responsible for a considerable delay, this is caused by the time-expensive segmentation techniques. The mechanisms and motors that manipulate the camera position are responsible for the other significant delay, the magnitudes of it depends on the particular devices characteristics.

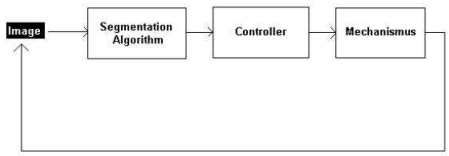

Figure 1. Visual tracking system representation.

A common way to solving this problem is to restrict the use of segmentation algorithms to relatively simple ones, use motors and expensive Hardware for image capture that assure a better dynamic behavior. However doing this, could limit the applications possibilities of the tracking systems. In this work we propose a neurofuzzy prediction algorithm to eliminate the delay problem. The neurofuzzy algorithm is able to predict in 6 frames up the dynamics of the target object, this time is enough for most of the applications, however this number could be improved without a great additional effort.

This work is organized in the following way, in section 2 the neurofuzzy model is described, section 3 explains the whole system implementation, finally in section 4 the obtained results are shown.

2. Adaptive Neuro-Fuzzy Inference system

In this section, we describe a class of adaptive network that are functionally equivalent to fuzzy inference systems. The propose architecture is referred to as ANFIS [1], which stands for adaptive network-based fuzzy inference system. We describe how to decompose the parameter set to facilitate the hybrid learning rule for ANFIS architecture representing both the Sugeno and Tsukamoto fuzzy models. The effectiveness of ANFIS with the hybrid learning is tested through one simulation example.

2.1. ANFIS architecture

For simplicity, we assume that the fuzzy inference system under consideration has two input x and y and output z. For a first-order Sugeno fuzzy model, a common rule set with two fuzzy if-then rules is the following:

Rule 1: If x is $A1$ and y is $B1$, then $f1 = p1x + q1y + r1$,

Rule 2: If x is $A2$ and y is $B2$, then $f2 = p2x + q2y + r2$,

The Fig. 2 illustrate the reasoning mechanism for this Sugeno model; the corresponding equivalent ANFIS architecture is shown in Fig. 3, where nodes of the same layer have similar functions, as described next.

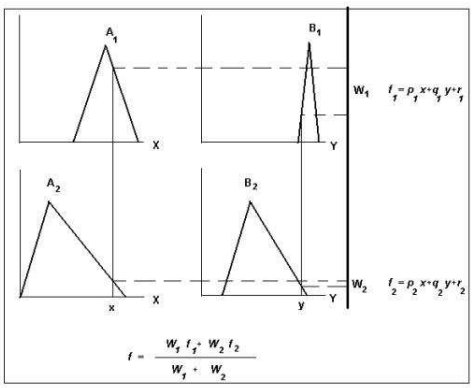

Figure 2. A two-input first-order Sugeno fuzzy model with two rules.

Layer 1. Every node I in this layer is an adaptive node with a node function:

$$O_{1,i} = mAi(x),\ for\ \ i = 1,2,\ \ or$$
$$O_{1,i} = mBi-2(y),\ for\ \ i = 3,4,$$

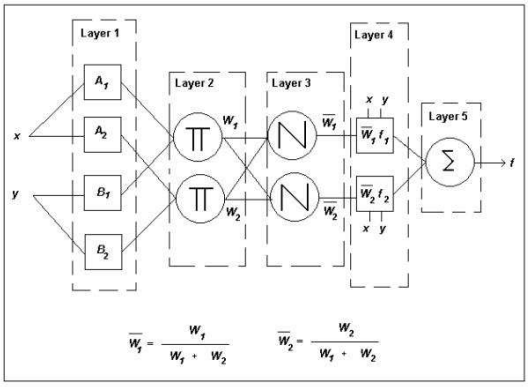

Figure 3. Equivalent ANFIS architecture.

where x (or y) is the input to node i and Ai (or $Bi-2$) is a linguistic label (such as "small" or "large") associated with this node. In other words, i is the membership grade of a fuzzy set A ($= A1, A2, B1$ or $B2$) and it specifies the degree to which the given input x (or y) satisfies the quantifier A. Here the membership function for A can be any appropriate parameterized membership function, such as the generalized bell function:

$$\mu_A(x) = \frac{1}{1 + \left|\frac{x-c_i}{a_i}\right|^{2b}}$$

where ai, bi and ci are the set parameters. As the values of these parameters change, the bell-shaped function varies accordingly, thus exhibiting various forms of membership for fuzzy set A. Parameters in this layer are referred to as premise parameters.

Layer 2. Every node in this layer is a fixed node labelled Π, whose output is the product of all the incoming signals:

$$O_{2,i} = \omega = \mu(x)\mu_{bi}(y), i = 1,2.$$

Each node output represent the firing strength of a rule. In general, any other T-norm operators that perform fuzzy AND can be used as the node function in this layer.

Layer 3. Every node in this layer is a fixed node labelled N. The ith node calculates the ratio of the ith rule's firing strength to the sum of all rule's firing strengths.

$$O_{3,i} = \varpi_i = \frac{\omega_i}{\omega_1 + \omega_2}, i = 1,2.$$

For convenience, outputs of this layer are called normalized firing strengths.

Layer 4. Every node i in this layer is an adaptive node with a node function

$$O_{4,i} = \varpi_i f_i = \varpi_i(p_i x + q_i y + r_i),$$

where ϖ_i is a normalized firing strength from layer 3 and pi, qi and ri are the set parameters of this node. Parameters in this layer are referred to as consequent parameters.

Layer 5. The single node in this layer is a fixed node labelled Σ, which computes the overall output as the summation of all incoming signals.

$$O_{5,i} = \sum_i \varpi_i f_i = \frac{\sum_i \omega_i f_i}{\sum_i \omega_i}$$

2.2. Hybrid Learning Algorithm

From ANFIS architecture shown in the figure 3, we observe that the values of the premise parameters are fixed, the overall output can be expressed as a linear combination of the consequent parameters. In symbols, the output f in the figure 3 can be rewritten as

$$f = \frac{\omega_1}{\omega_1 + \omega_2} f_1 + \frac{\omega_2}{\omega_2 + \omega_2} f_2$$
$$= \varpi_1(p_1 x + q_1 y + r_1)\varpi_2(p_2 x + q_2 y + r_2)$$

$$= (\varpi_1 x p_1 + (\varpi_1 y) q_1 + (\varpi_1) r_1 + (\varpi_2 x) p_2 + (\varpi_2 y) q_2 + (\varpi_2) r_2,$$

which is linear in the consequent parameters p_1, q_1, r_1, p_2, q_2 and r_2. From this observation, we have: S=set of total parameters, $S1$=set of premise (nonlinear) parameters, $S2$=set of consequent (linear) parameters.

The learning algorithm for ANFIS is a hybrid algorithm which is a combination between gradient descent and least-squares method. More specifically, in the forward pass of the hybrid learning algorithm, node outputs go forward until layer 4 and the consequent parameters are identified by the least-squares method. In the backward pass, the error signals propagate backward and the premise parameters are updated by gradient descendent . The Table 1 summarizes the activities in each pass.

The consequent parameters are identified optimal under the condition that the premise parameters are fixed. Accordingly, the hybrid approach converges much faster since it reduced the search space dimensions of the original pure backpropagation method.

Table 1. Activities in each pass

	Forward pass	Backward pass
Premise parameters	fixed	gradient descendent
Consequent parameters	least-squares estimator	fixed
Signals	node outputs	error signals

3. Implementation

3.1. Description

For this work we solve the tracking problem of a soccer ball whose movements can be strongly or violent, the objective therefore is to maintain the visual contact of the object in the image frame. The mechanism which manipulates the camera position is constituted of two independent RC-servo motors. In this section we explain the development and content of the blocks which integrate the tracking system (Fig. 1).

3.2. Segmentation algorithm and localization

The object to be segmented is characterized to present a pattern color, for this reason a segmentation algorithm with the capacity to segment color was chosen. Thus a simple change of color model and image threshold would be enough for the ball segmentation. The Fig. 4 shows the object to be segmented.

Figure 4. Object to be segmented.

The image is transformed of the RGB model to the HSV model, which is more appropriate for the color segmentation, later the image is divided in its characteristic planes H, S and V. From those planes we take only the S plane and then we apply a threshold value of 220. The Fig. 5 shows the result of this process.

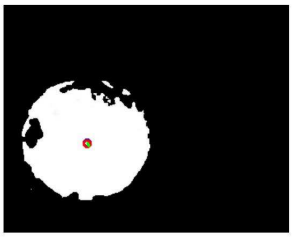

Figure 5. Segmented Object.

Once the object is located, then we can use the moment method to identify the centroid. The used equations are:

$$M_{00} = \sum_x \sum_y I(x,y),$$

$$M_{10} = \sum_x \sum_y xI(x,y), M_{01} = \sum_x \sum_y yI(x,y),$$

$$x_c = \frac{M_{10}}{M_{00}}, y_c = \frac{M_{01}}{M_{00}},$$

where M_{00} represents the zero degree moment while $M_{!0}$ and M_{01} means the first degree moments of x and y respectively while xc and yc represents the center coordinates.

3.3. The Controller

For this work we use a PID controller for each axis movement (It can also be used a fuzzy controller as is described in [2]). The objective of the visual tracking is to maintain the target object inside of the image frame, therefore the controller's input would be the existent differences between the object localization point and the image central point. The Fig. 6 shows this process.

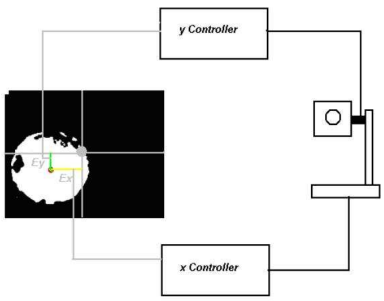

Figure 6. Controller configuration.

The above configuration, due the existent delay works poorly. We propose in this work to eliminate this problem incorporating a neurofuzzy predictor system of the object dynamics. Thus the system is modified as is shown in the Fig. 7.

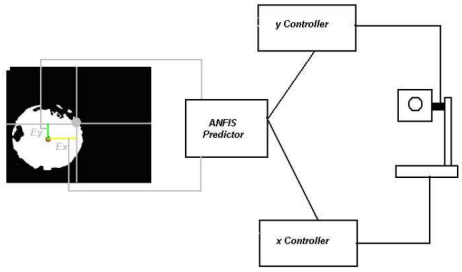

Figure 7. Modified system.

Here the ANFIS system predicts the object dynamics 6 times up, therefore the inherent delays of the system are compensated.

3.4. Predictor design

The predictor ANFIS will be divided in two predictors which correspond to each axe movement x and y. For the predictor design, the ball localization is saved in a file, this is achieved moving the ball to the speeds and accelerations desired. Thus, we will have the necessary information for the system training. The Fig. 8 shows the saved movements.

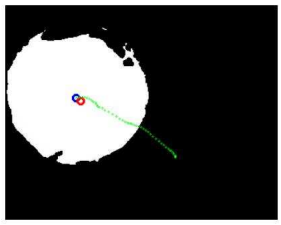

Figure 8. Saved movements.

For the ANFIS training, we will use the tools implemented in matlab [3]. The data stored in the file are assigned to a matlab vector, starting from here we can use the commands of the fuzzy logic toolbox. In this part we present the training steps of a movement axis being the training of the other one exactly identical. If the map of movements generated by the ball corresponds to the Fig. 9

and Fig. 10, then we can separate the generated movements in the x axis, as shows Fig. 11.

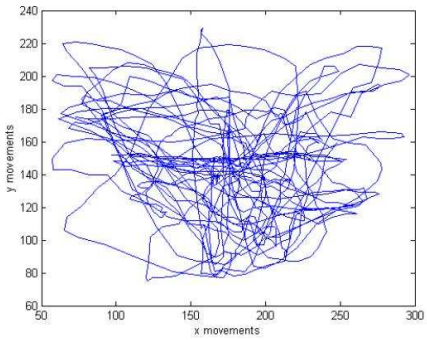

Figure 9. x and y movements.

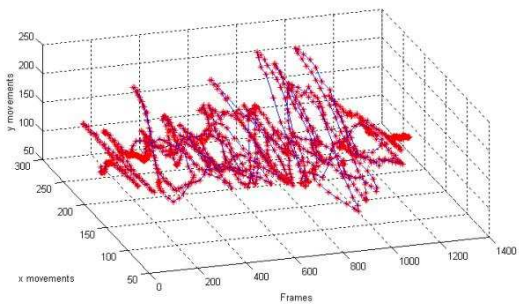

Figure 10. x and y movements in the space.

The learning results can be summarized as the final membership functions and the rules equations of the Sugeno inference system. Fig. 11 shows the membership functions of the input x(t-18). The previous procedure is repeated for the y axis movements.

The complete system was coded in C++ and tested on a PCx86 at 900MHz with 128Mbytes RAM, operating in real time on an image of 352x288 pixel using an USB-Webcam.

4. Results

We have successfully developed, implemented and tested a neurofuzzy system for predicting the motion of a target object. The prediction compensates the system delay

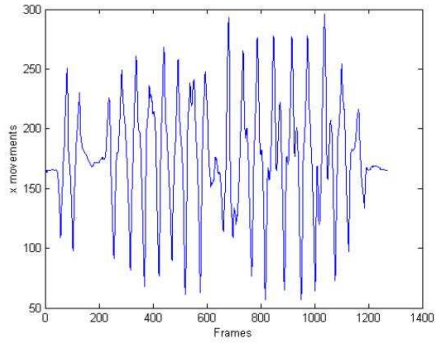

Figure 11. x movements.

and thus allows precise and fast motion control. To demonstrate the prediction effect on the system behavior, we have tested the tracking object in high velocity. It performs very well and we have nearly eliminated the influence of the delay on the system. The Fig. 12 shows the predictor performance for x axis.

The system performance is very good, allowing to predict the movement of the target object with a minimum error.

References

[1] Jang S. R., Sun C. T., Mitzutani E. Neurofuzzy and Soft Computing, Prentice Hall, New York, (1998).

[2] Cuevas E., Zaldivar D.,Rojas R. Intelligent Tracking. Technical Report B-13-03, Freie Universitt Berlin, Germany (2003).

[3] Fuzzy logic Toolbox, Mathworks, New York(1999).

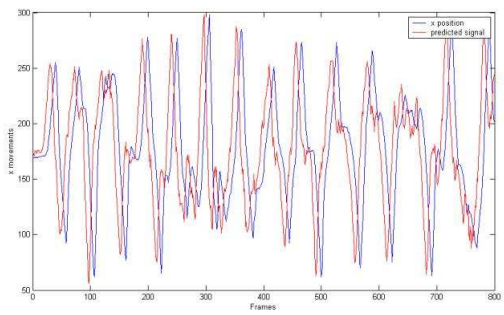

Figure 12. predictor performance for x axis.