A Tool for the Development of Robot Control Strategies

Luiz Carlos Figueiredo[1], Gilcésar Ávila[1], Francisco Reinaldo[1,2], Rui Camacho[2], Demétrio R. Magalhães[1], Luís Paulo Reis[3]

[1] UnilesteMG - Centro Universitário do Leste de Minas Gerais
GIC - Grupo de Inteligência Computacional
Av. Presidente Tancredo Neves n. 3500, Bairro Universitário, CEP 35170-056
Coronel Fabriciano, MG, Brasil
{figueiredo,gilcesar,demetrio}@unilestemg.br
[2] FEUP - Faculdade de Engenharia da Universidade do Porto
LIAAD - Laboratório de Inteligência Artificial e Apoio a Decisão
Rua Dr. Roberto Frias, sn, 4200-465
Porto, Portugal
rcamacho@fe.up.pt
[3] FEUP - Faculdade de Engenharia da Universidade do Porto
LIACC - Laboratório de Inteligência Artificial e Ciência de Computadores
Rua Dr. Roberto Frias, sn, 4200-465
Porto, Portugal
{reifeup,lpreis}@fe.up.pt

Abstract

In this paper we report as the development of a tool in to develop and set control strategies as a fast and easy way. Additionally, a tricycle robot with two traction motors was built to test the strategies produced with the tool. Experimental tests have shown an advantage in the use of such tool.
Key-words: Development Tools, Behaviour, Fuzzy, Robot.

1 Introduction

Nowadays, autonomous robots has received the main focus on research centres at different institutions worldwide. This field of research can be an harmonic mixture of many disciplines such as control systems, instrumentation, image and signals filtering, and micro controllers. A challenging topic in this study is the development of strategies and devices to obtain movements oriented by control systems.

Unfortunately control systems have presented limitation of movements in autonomous robots, making a challenge and an unsolved problem [5]. In order to solve these limitations, Murray [7] has been working in parallel with non-linear controllers and Artificial Intelligence. In this sense, a lot of research has being made in order to solve the common problems of how to achieve a balance in linear controllers. The main idea is to develop a control strategy that equilibrates the system on origin. In this paper, we propose another approach of that proposed by Murray. We develop a tool for non expert users to be able to easily plan control strategies, and a robot to test them. In fact, a fuzzy controller have been implemented to supply the need of control strategies.

The rest of the paper is divided in six sections. Section 2 introduces the project, the hardware employed, and the mathematical model of fuzzyfication. Section 3 explains the Fuzzy controller inspired on the work of [6]. Section 4 comments the classes used to interface with traction motors and Fuzzy controller. Section 5 present the experiments and discuss some results. In Section 6, we draw some conclusions and point future directions for this work.

2 Project Description

The project consists to develop a robot and a control strategy module for the robot. For this purpose, we have assembled a robot to be sufficiently autonomous to navigate in a dynamic environment with obstacles moving around. The robot structure is similar to the tricycle robots were employed in football championships. After many proposals, we conclude that the current agent structure produces fast responses from control and trajectory algorithms.

Basically the robot structure is composed of a micro controller, a free wheel and two traction motors, and a camera. The robot structure is as follows.

Micro-controller: interchanges communication signals between an external CPU and two traction motors;

Traction motors: moves the robot to a given direction. These motors performs movements by a sequence of electric pulses in its reels. These signals, coming from the micro controller, represents the robot movements. Both traction motors can perform movements on the same direction, whether the robot needs to walk on a right line. Different velocities but in the same direction can also produce curves over that wheel with low velocity. Performing rotations in different directions, the robot will rotate in its central vertical axis. With these features and a camera, the robot can accomplish a complex task of moving from point A (x, y) to point B (x_1, y_1);

Camera: captures information from the environment and sends it to the CPU. Thus, the robot can navigate at the environment and fix its trajectory. Based on Kalman filters, we have improved the trajectories routine of the robot [4].

2.1 Mathematic tricycle model

The mathematic model of tricycle is given by Figueiredo as follows [1]:

$$\begin{aligned} x &= v\cos(\Theta) \\ y &= v\sin(\theta) \\ \Theta &= w \end{aligned} \tag{1}$$

The robot coordinates (x, y) corresponds to the centre point of axis that connects both traction motors (wheels) and θ denotes the direction angle was obtained from axis x. The signal of control, such as v and w, represents the linear and angular robot velocity, respectively. The signal of control is also known as Canonic Cartesian [2] and was obtained in this manner:

$$\begin{aligned} x_1 &= \Theta \\ x_2 &= x\cos\Theta + y\sin\theta \\ x_3 &= x\sin(\theta) + y\cos(\Theta) \end{aligned} \tag{2}$$

Consequently, a canonic formula (3) is reached. This last formula[7] allow us to balance the robot in its axis [5].

$$\begin{aligned} x_1 &= u_1 \\ x_2 &= u_2 \\ x_3 &= x_2 u_1 \end{aligned} \tag{3}$$

where

$$\begin{aligned} u_1 &= w \\ u_2 &= v - x_3 w \end{aligned} \tag{4}$$

Thus, we can find an approximate robot position by performing (5):

$$\begin{aligned} x_{new} &= x_{old} + vt\cos\Theta \\ y_{new} &= y_{old} + vt\sin\Theta \\ \Theta_{new} &= \Theta_{old} + wt \end{aligned} \tag{5}$$

The calculus presented above is associated with uncertain values obtained from the velocities v and w, as well as eventual wheels slithering or temporary wheels blocking. It is worth noting that many "errors" can be accumulated with time, and a bad behaviour could occur. In this case, whether different rate sampling is obtained, we use Kalman filter to minimise the error rate [4].

3 Fuzzy Controller

In accordance with Figueiredo [3], the Fuzzy controller project consists in:

- defining the universe of discourse of linguistic variables, error rate, error variation, and output controller variation;
- defining the amount of primary terms and relevancy degrees of diffuse subsets that represents each terms;
- determining some rules that compose control algorithm;
- determining parameters of the project, such as inference methods, what logic will be employed, defuzzification form, and controller performance.

In this work, we define the primary terms by the use of a trapezoid function. The rule used in the fuzzyfication was:

```
 IF ((x - p1) OR (p2 - x))
 THEN f(x) := 0;
 ELSE
 f(x) := min(Delta1*alfa1,
Delta2*alfa2,
superior bound);
```

Therefore, the defuzzification process uses the Singleton method to be simple. Singleton is an output function with a communication degree represented by an unique vertical line. This vertical line represents the gravity centre of the linguistic term to achieve the maximum strength. If a Singleton intercepts the axis x in an unique local, the calculus of gravity centre is reduced only to the calculus of a balanced average of x to each Singleton and its strength degree.

4 Assembling the robot

The robot development was divided in two main parts: Robot Controller Classes, and Fuzzy Controller Classes.

4.1 Robot Controller Classes

The robot receives commands from a external CPU by a serial port RS232. This interface with traction motors board is provided by three classes, as follow.

First, **TStepControl class** encapsulates the commands of Solbet StepControl board. This class shares methods in a high level to control the robot. The methods implemented are specialised commands such as:

- TurnOn method: starts the traction motor movements

- TurnOff method: stops the traction motor movements

- Sterring method: receives angles from the CPU to change the robot direction. The robot can turn left or turn right in a specific positive or negative angle

Second, the **TCommPortStrategy class** defines an abstract class that has a common interface that supports several algorithms to access the serial port. In this sense, the TStepControl class will not need to be updated and recompiled for each change submitted to the routine, allowing optimisation and high portability.

Last, the **TAPICommPortStrategy** encapsulates the functions used to access the serial port by Windows API. This class is inherited from TCommPortStrategy.

4.2 Fuzzy Controller Classes

Fuzzy controller uses the current positions of the robot to calculate its trajectory.

In order to offer simple instructions to implement a Fuzzy controller, we have constructed some specialised interconnected classes that working together to perform some tasks. A brief description of each class with respective methods is presented below.

TMFShape class: defines an abstract class that has an member function shape interface for its offspring classes. Many sorts of member functions can be used to define the Fuzzy primary terms. The methods used are:

- Type(): returns specific Fuzzy functions, such as "Triangle", "Trapezoid", and others

- Centre(): returns the gravity centre value of picture

- Area(): returns the area value of picture

- Y(): returns a fuzzy value of X

Features:

- X: it is used to set or to obtain a value of X;

TMF class: links by reference a TMFShape instance. This class is a member function.

TVariable class: represents the input/output variables. In addition this class is composed by a set of member functions that defines the universe of speech. The main methods are:

- TVariable: defines a constructor that set all elements and universe of speech
- AddMF: adds a member function
- RemoveMF: removes a member function
- MFCount: returns a value of how many functions are in the variable
- MFByName: calls a member function

TRule class: represents a rule that will be used in the context. The methods are showed as follow:

- TRule: defines a constructor to set both input and output member functions. A rule example is represented as follows:

```
 IF ( ''member function n'' AND
''member function m'' )
 THEN ''member function z'';
```

TFuzzy class: both integrates methods and behaviours of other classes. This class receives values from and send values to robot control modules. The methods are:

- AddRule : adds a new rule to the rules database
- RemoveRule: removes a new rule from the rules database

Features:

- Position: receives the position values of x, y
- Angle: receives the robot angle in accordance with its abscissa axis
- Steering: sends the direction that a robot will do. It is a defuzzificated value
- Input1: receives the first input value
- Input2: receives the second input value
- Output: receives the output value

5 Experiment and Results

The main idea is measure our control strategy with Kosko's Fuzzy controller. Thus, three instances of TVariable class were created in order to develop a control strategy. The first instance represents all primary terms (two input variables, and an output variable) linked with name and universe of speech of each one of them. In the second instance, function members were created to populate the TFuzzy instance. The third instance, new rules are added into the rules database by AddRule method of TFuzzy class. The rules have the same structure proposed by [6], which is present below:

Figure 1 presents the robot position X on a flat surface.

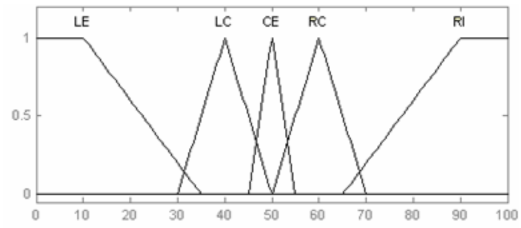

Figure 1: Input variable: Position. LE: Left, LC: Left Centre, CE: Centre, RC: Right Centre, and RI: Right.

In Figure 2, the robot angle X on flat surface is presented.

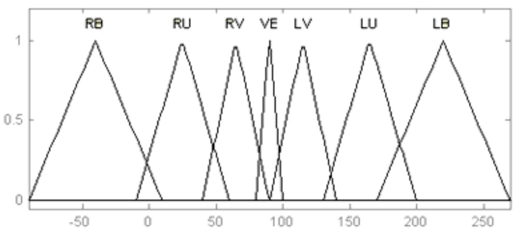

Figure 2: Input variable: Position. RB: Right Below, RU: Right Upper, RV: Right Vertical, VE: Vertical, LV: Left Vertical, LU: Left Upper, and LB: Left Below.

Figure 3 shows the guidance is measured by Fuzzy set.

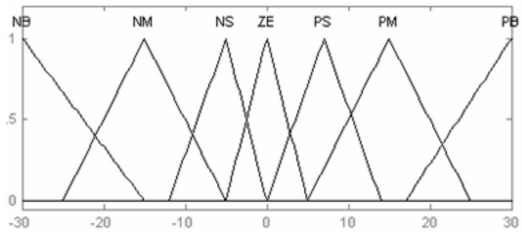

Figure 3: Output variable: Turn. NB: Big Negative, NM: Medium Negative, NS: Small Negative, ZE: Zero, PS: Small Positive, PM: Medium Positive, PB: Big Positive.

The figures 2, 3 and 4 show member functions to each variable of Fuzzy set. As it can be seen above, functions closer the centre of the variable were fine tuned than ones on the was achieved by out of centre.

We have compared the trajectory rules obtained by our controller with the Kosko's Fuzzy controller. After forty simulation cycles, statistical tests have concluded our proposed control strategy have achieved seven percent more control than Kosko's controller. This represents significant results given the fact that only cheaper devices were used, instead of those used in football championship.

Further experiments have been started in order to use Genetic Algorithm to fine tune the best robot parameters, such as position, linear and angular velocity.

6 Conclusions

In this paper, we have described a flexible development tool that allow users to implement several control strategies, measurement and communication devices.

The Fuzzy controller implemented was compact, and the rules produced can be easily changed as well as the primary terms. The results can be compared with theoretic studies and simulations in MatLab. The lack of an absolute measure position device can compromiser the performance of the robot by accumulation of error. The use of secondary measure devices can obtain a better performance.

References

[1] FIGUEIREDO, L. C. Modelagem matemática de sistemas de rastreamento de veículos. *Revista DOXA 7* (2002), 17–28. Coronel Fabriciano-MG.

[2] FIGUEIREDO, L. C. & JOTA, F. G. A switching time-varying and time invariant controller to stabilize nonholonomic systems. In *XIV Congresso Brasileiro de Automática - CBA*, Natal - RN, 2002.

[3] FIGUEIREDO, L. C. & TEIXEIRA, R. A. Implementação de um controlador fuzzy em clp. In *II Congresso Mineiro de Automação, V Simpósio Regional de Instrumentação*, Belo Horizonte - MG, 1998, p. 73–79.

[4] FIGUEIREDO, L.C. & JUSTINO, M. Robustez em medição: Filtro de kalman aplicado a veículos autônomos. *Revista DOXA 6* (2001), 75–88.

[5] KOLMANOVSKY, I. & MCCLAMROCH, N. H. Developments in nonholonomic control problems. In *IEEE Trans. On Control Systems*, 1995, p. 20–36.

[6] KOSKO, B., Ed. *Neural Networks and Fuzzy Systems – A Dynamical Systems Approach to Machinne Intelligence*. Prentice Hall, Englewood Cliffs, 1991. ISBN 0136114350.

[7] MURRAY, R. M.; LI, Z. & SASTRY, S. *A mathematical introduction to robotic manipulation*. CRC Press LLC, 1994. ISBN 0849379814.