VIMS'99, IEEE Workshop on Virtual and Intelligent Measurement Systems, Venice, Italy, May 21, 1999.

Fuzzy Controller for a Hexaped Robot

Ziad T. Sakr (1), Emil M. Petriu (1), and Peter Wide (2)

(1) SITE - School of Information Technology and Engineering, University of Ottawa, Canada,
(2) Applied Autonomous Sensor Systems Laboratory, Orebro University, Sweden,

Abstract: *This paper presents a fuzzy controller for a six legged walking robot. Contact sensors, one on each leg, inform the controller when the legs actually touch the ground allowing it to adapt to uneven floor surfaces .*

1. Introduction

Classical control is based on a detailed I/O function OUTPUT= F (INPUT) which maps each high-resolution quantization interval of the input domain into a high-resolution quantization interval of the output domain as illustrated in Figure 1. Finding a mathematical expression for this detailed mapping relationship F may be difficult in many applications, if not impossible.

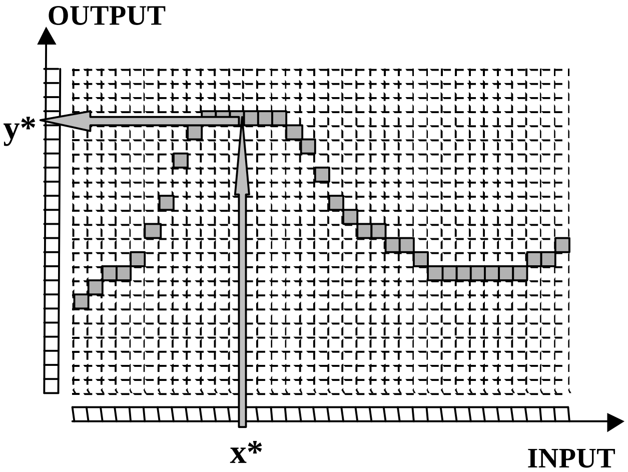


Figure 1. Classic control implements a detailed I/O mapping relationship

The fuzzy logic control, [1]-[4], provides a non-analytic alternative to the classical analytic control theory. The behavior of a fuzzy controller can be described with simple *if-then* relations based on very low-resolution models able to incorporate empirical (i.e. not too "certain") engineering knowledge. In many applications, this leads to a simpler solution in less design time. FLCs have found many practical applications in the context of complex ill-defined processes that can be controlled by skilled human operators: water quality control, automatic train operation control, nuclear reactor control, automobile transmission control, etc.

As shown in Figure 2, a "fuzzy logic controller" (FLC) consists of three major components: input interface, inference mechanism, and output interface.

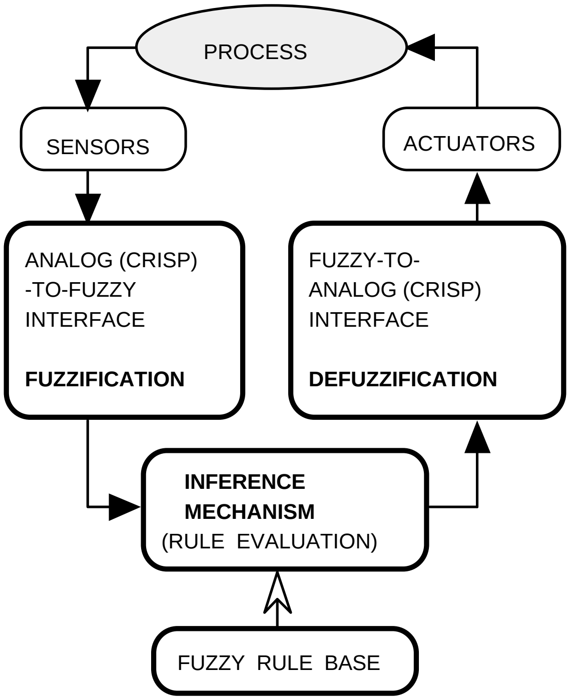


Figure 2. The structure of a fuzzy logic controller

The domain of each input- or output-variable is partitioned into a finite number of overlapping fuzzy sets. Each fuzzy set is characterized by a membership function which assigns to each point in the domain a real number between 0 and 1 representing the degree of membership of that point to the given fuzzy set.

A FLC implements a composite I/O function that maps each very low-resolution quantization interval of the input domain into a very low resolution fuzzy-quantization interval, [5], of the output domain as shown in Figure 3. As there are usually only 7 or 9 fuzzy-quantization intervals covering the input and output domains, the mapping relationship can be very easily expressed using the *if-then* inference formalism. The overlapping of these fuzzy domains and their linear membership functions will eventually allow a rather high-resolution I/O function to be achieved between crisp input and output variables.

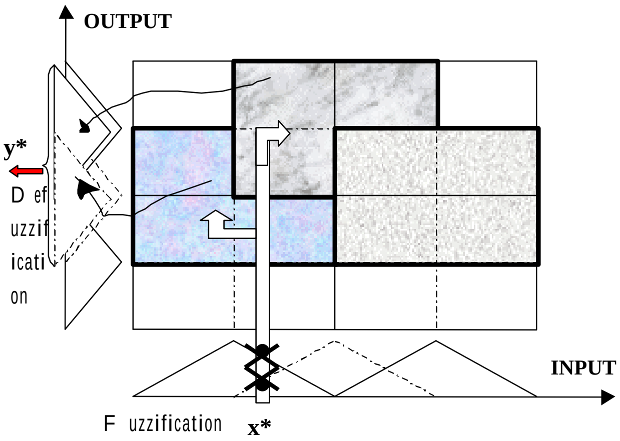


Figure 3. FLC implements a composite low resolution I/O mapping relationship

This paper discusses the design of a FLC for the movement of a hexaped (six–leg) robot. Since a hexaped robot's movement is much harder to control than a four-legged robot, especially on rough surfaces, a FLC has been designed to keep that movement balanced and smooth. This controller should control the robot's forward movement while preventing it from tipping over.

2. Problem Analysis

Figure 4 shows a top view of the hexaped robot and Figure 5 shows the front view of the three pairs of legs. Legs 1 and 2 are connected together in tandem through a horizontal bar. Servo 6 controls their movement (back and forth) in the horizontal plan and servo 5 controls their movement (up and down) in the vertical plan.

Legs 3 and 4 are connected together through a horizontal bar. They move in tandem under the control of servo 4 in the horizontal plan and of the control servo 3 in the vertical plan. Legs 5 and 6 are connected together through a horizontal bar. They move in tandem controlled by servo 2 in the horizontal plan and by servo 1 in the vertical plan.

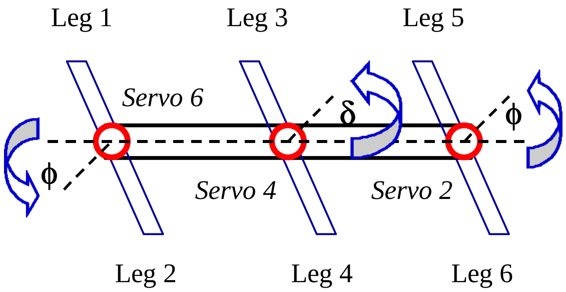


Figure 4. Top view of the hexaped robot

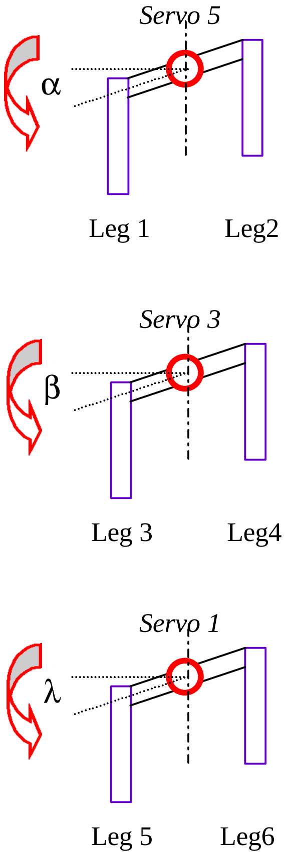


Figure 5. Front view of the robot legs

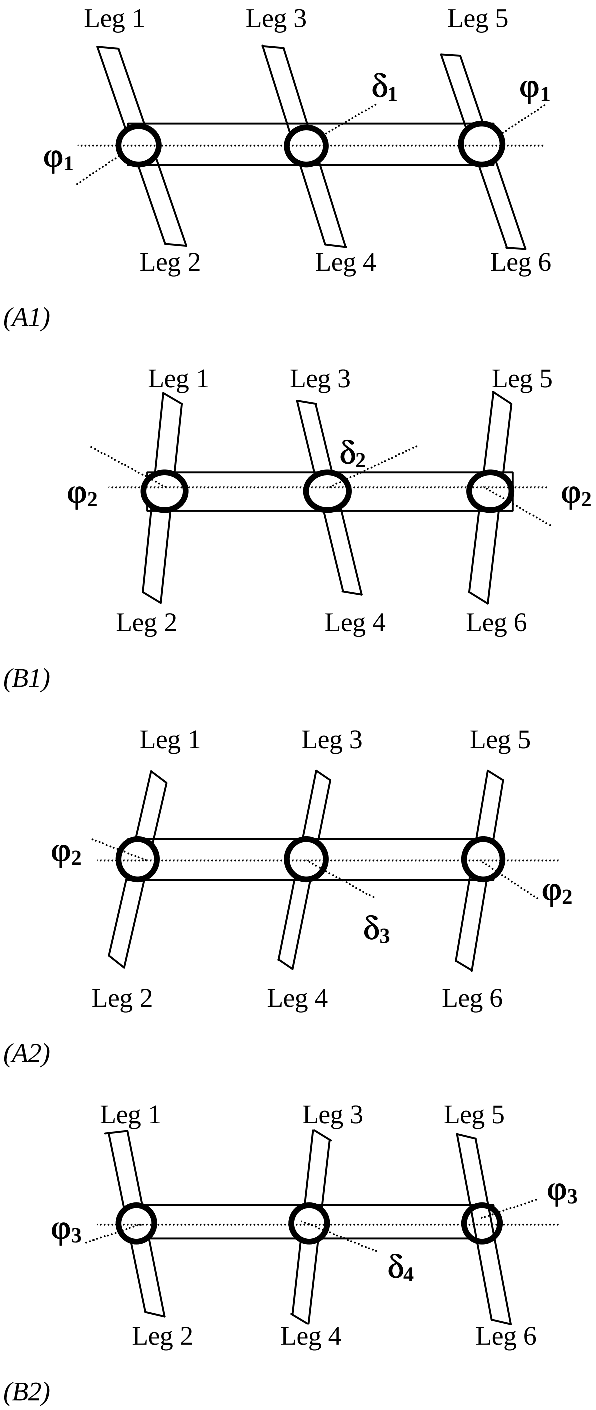


Figure 6. Robot movement analysis

Figure 6 illustrates a four-phase strategy used to move the robot forward. We have to make sure that the right legs are used and that they are positioned at the proper angles to prevent the robot from tipping over.

(i) If legs 1 and 2 are rotated to position $\varphi_1 < 0$, then legs 5 and 6 should also be rotated to φ_1, and legs 3 and 4 should be adjusted to an angle δ_1 (where $\delta_1 < 0$) in such a way that, the distance between leg 1 and leg 4 is the same as that between leg 4 and leg 5 as shown in Figure 6-A1. This adjustment of legs 3 and 4 prevents the body from tipping over. Leg 1 can then be moved to the ground by an angle of α in the positive direction, leg 4 by an angle of β in the negative direction, and leg 5 by an angle of λ in the negative direction.

(ii) The robot will now advance by rotating legs 1 and 2, and legs 5 and 6 to position $\varphi_2 > 0$ and rotating legs 3 and 4 to position $\delta_2 < \delta_1$, as shown in Fig. 6-B1.

(iii) Now, in order to prepare for the next step, all legs should go back down to the ground. If legs 1 and 2, and legs 5 and 6 are positioned at $\varphi_2 > 0$, legs 3 and 4 should be adjusted to position $\delta_3 > 0$ in such a way that the distance between legs 2 and 3 is the same as that between legs 3 and 6, as shown in Fig. 6-A2. Leg 2 can then be moved to the ground by an angle of α in the negative direction, leg 3 by an angle of β in the positive direction, and leg 6 by an angle of λ in the positive direction.

(iv) The robot can now make another forward step by moving legs 1 and 2, and legs 5 and 6 to position $\varphi_3 < 0$ and moving legs 3 and 4 to position $\delta_4 > \delta_3$, as shown in Fig. 6-B2.

This four-phase cycle can be repeated as many times as desired until the robot moves forward to destination. However, if the ground is not perfectly flat then an adaptive mechanism is needed to allow the robot to *balance* itself and continue advancing. This can be done as shown in the next chapter by using the feedback from the contact sensors at the bottom of its legs.

3. Fuzzy Logic Controller

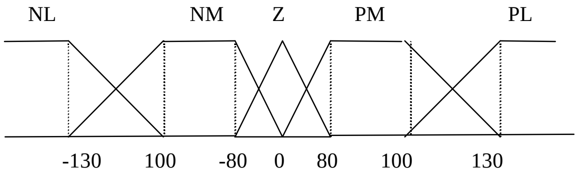
The developed FLC uses four input fuzzy variables shown in Figure 7: *Front(B)* = $\alpha 1$, *Middle(B)* = $\beta 1$, *Rear (B)* = $\lambda 1$ and *Body(B)* = $\varphi 1$ which are the legs and robot's body positions when all legs are resting on the surface ("B" = Before). There are also four output fuzzy variables shown in Figure 8: *Front(A)* = $\alpha 2$, *Middle(A)* = $\beta 2$, *Rear(A)* = $\lambda 2$, and Body(A) = j *2,* which are positions to be taken for the robot to move forward by shifting the body to position $\varphi 2$ ("A" = After).

There are 39 *if-then* rules as we taking into consideration that only one leg might step on an object. This reduces the problem from the 375 rules ($5\alpha * 5\beta * 5\lambda * 3\varphi$) if all possible cases where taken.

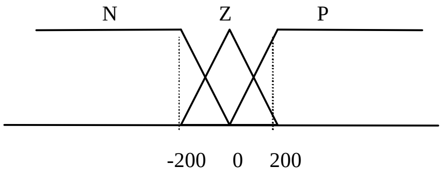
3.1. Membership functions

The membership functions of the input variables *Front(B), Middle(B), Rear(B)* and *Body(B)* are shown in

Figure 7. The output variables *Front(A), Middle(A), Rear(A),* and *Body(A)* are singletons as shown in Figure 8.

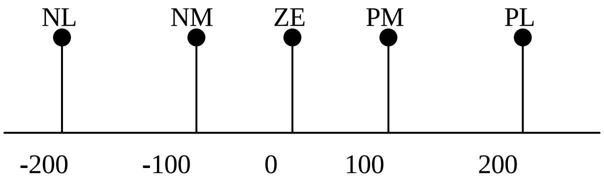


a) Input variables *Front(B), Middle(B), and Rear(B)*

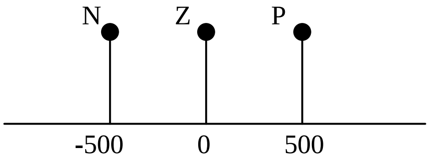


b) Input variable *Body(B)*

Figure 7. Input membership functions



a) Output variables *Front(A), Middle(A), and Rear(A)*



b) Output variable *Body(A)*

Figure 8. Output membership functions

These singleton output variables which allow the robot to walk in the best way possible, were chosen through experimentation.

3.2. If-then rules

The rules that map the various fuzzy values of the inputs Front(B), Middle(B), Rear(B), and Body(B) to the outputs Front(A), Middle(A), and Rear(A) are represented in matrix form as shown in Figures 9, 10 and 11.

Body(B) *Front(B)*	N	Z	P
NL	NM/NL/NL/P	NL/ZE/ZE/Z	NL/PL/PL/N
NM	ZE/NM/NM/P	NM/ZE/ZE/Z	NL/PL/PL/N
ZE	PL/NL/NL/P	ZE/ZE/ZE/Z	NL/PL/PL/N
PM	PL/NL/NL/P	PM/ZE/ZE/Z	ZE/PM/PM/N
PL	PL/NL/NL/P	PL/ZE/ZE/Z	PM/PL/PL/N

Figure 9. If-then rules for *Front(A)/ Middle(A)/ Rear(A)/ Body(A),* with *Middle(B)* = ZE and *Rear(B)* = ZE

Body(B) *Middle(B)*	N	Z	P
NL	PL/NL/NL/P	ZE/NL/ZE/Z	NL/NM/PL/N
NM	PL/NL/NL/P	ZE/NM/ZE/Z	NM/ZE/PM/N
ZE	PL/NL/NL/P	ZE/ZE/ZE/Z	NL/PL/PL/N
PM	PM/ZE/NM/P	ZE/PM/ZE/Z	NL/PL/PL/N
PL	PL/PM/NL/P	ZE/PL/ZE/Z	NL/PL/PL/N

Figure 10. If-then rules for *Front(A/ Middle(A)/ Rear(A)/ Body(A),* with *Front(B*) = ZE and *Rear(B)* = ZE

Body(B) *Rear(B)*	N	Z	P
NL	PL/NL/NL/P	ZE/ZE/NL/Z	NL/PL/NM/N
NM	PL/NL/NL/P	ZE/ZE/NM/Z	NM/PM/ZE/N
ZE	PL/NL/NL/P	ZE/ZE/ZE/Z	NL/PL/PL/N
PM	PM/NM/ZE/P	ZE/ZE/PM/Z	NL/PL/PL/N
PL	PL/NL/PM/P	ZE/ZE/PL/Z	NL/PL/PL/N

Figure 11. If-then rules for *Front(A)/ Middle(A)/ Rear(A)/ Body(A)* with *Front(B)* = ZE and *Middle(B)* = ZE

3.3. Defuzzification

The defuzzification process is used to convert fuzzy outputs to crisp numbers. The values of the output variables were defined as singletons to simplify the defuzzification process and the robot movement. The crisp values are calculated as follows:

In the case of the fuzzy output *Front(A)* controlling the position of legs 1 and 2, while the rest of the legs are resting on the ground (zero position) and the fuzzy output for the body is *Body(B),* the crisp value for the angle α is:

Crisp_α =[min(μFront(B1), μBody(B1)) * Front(A1)] +
[min(μFront(B1), μBody(B2)) * Front(A2)] +
[min(μFront(B2), μBody(B1)) * Front(A3)] +
[min(μFront(B2), μBody(B2)) * Front(A4)]} /
{[min(μFront(B1), μBody(B1))] + [min(μFront(B1), μBody(B2))] + [min(μFront(B2), μBody(B1))] +
[min(μFront(B2), μBody(B2))]}

where μFront(Bx) and μBody(Bx) determine the membership degree of the fuzzy output for legs 1 and 2, and body position respectively. Front(Ax), Middle(Ax), Rear(Ax) and Body(Ax) are singleton fuzzy outputs.

Similar formulas can also be used for the defuzzification of the fuzzy outputs for the legs 3 and 4, or the legs 5 and 6:

Crisp_β =[min((μFront(B1), (μBody(B1))*Middle(A1)]+
[min((μFront(B1), (μBody(B2)) * Middle(A2)] +
[min((μFront(B2), (μBody(B1)) * Middle(A3)] +
[min((μFront(B2), (μBody(B2)) * Middle(A4)]} /
{[min((μFront(B1), (μBody(B1))] + [min((μFront(B1), (μBody(B2))] + [min((μFront(B2), uBody(B1))] +
[min((μFront(B2), (μBody(B2))]}

Crisp_λ ={[min((μFront(B1), (μBody(B1)) * Rear(A1)] +
[min((μFront(B1), (μBody(B2)) * Rear(A2)] +
[min((μFront(B2), (μBody(B1)) * Rear(A3)] +
[min((μFront(B2), (μBody(B2)) * Rear(A4)]} /
{[min((μFront(B1), (μBody(B1))] +
[min((μFront(B1), (μBody(B2))] + [min((μFront(B2), (μBody(B1))] + [min((μFront(B2), (μBody(B2))]}

The crisp value for for the body position φ is:

Crisp_φ ={[min((μFront(B1),(μBody(B1))*Body(A1)] +
[min((μFront(B1), (μBody(B2)) * Body(A2)] +
[min((μFront(B2), (μBody(B1)) * Body(A3)] +
[min((μFront(B2), (μBody(B2)) * Body(A4)]} /
{[min((μFront(B1), (μBody(B1))] +
[min((μFront(B1), (μBody(B2))] + [min(uFront(B2), (μBody(B1))] + [min((μFront(B2), (μBody(B2))]}

4. Experimental Setup

An experimental hexaped robot was built using Robix RCS-6 robot kit components. Six switches were used as contact sensors, one on each leg, allowing the controller to take the appropriate action when the legs touch the ground or a solid object.

The fuzzy controller was implemented as a C program. The developed FLC has been tested for various initial conditions and it has fully met the expectations.

The first experiments have shown that the robot leg positions have an absolute error of 2 units caused by the sensors placed at the bottom of each leg. This error was corrected by making a few adjustments to the fuzzy controller.

The robot's movement was limited by considering only 39 *if-then* rules, where only one leg is allowed to step on a solid object in its way, while the rest have to lie on a flat surface. But the movement can be enhance and be more interesting if all the 375 *if-then* rules were consider, the robot can then move on any surface.

References

[1] L.A. Zadeh, "Fuzzy algorithms," *Information and Control,* vol. 12, pp. 94-102, 1968.

[2] L.A. Zadeh, "A rationale for fuzzy control," *J. Dynamic Syst. Meas. Control,* vol.94, series G, pp.3-4, 1972.

[3] E.H. Mamdani and N.S. Assilian, "A case study on the application of fuzzy set theory to automatic control,"*Proc. IFAC Stochastic Control Symp*, Budapest, 1974.

[4] E.H. Mamdani, "Twenty years of fuzzy control: Experiences gained and lessons learnt," *Fuzzy Logic Technology and Applications*, (R.J. Marks II, Ed.), IEEE Technology Update Series, pp.19-24, 1994.