Humanoid Android Sub-system Capstone Project - Fuzzy Control

Zhong Nan Zhang

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2014-97
http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-97.html

May 16, 2014

University of California, Berkeley College of Engineering

MASTER OF ENGINEERING - SPRING 2014

Electrical Engineering and Computer Science

Physical Electronics

Humanoid Android Sub-system Capstone Project

Fuzzy Control

Zhong Nan 'Julian' Zhang

This **Masters Project Paper** fulfills the Master of Engineering degree requirement.

Approved by:

1. Capstone Project Advisor:

Signature: __________________________ Date ____________

Print Name/Department: Dr. Donald Wroblewski/Fung Institute for Engineering Leadership

2. Faculty Committee Member #2:

Signature: __________________________ Date ____________

Print Name/Department: Prof. Ruzena Bajcsy/EECS

Humanoid Android Subsystem Capstone Project

Fuzzy Control

Abstract

This paper focuses on the development of fuzzy control for the robotic arm in the Humanoid Android Subsystems Capstone Project. Fuzzy control is artificially intelligent control. It not only fulfills the role of basic control that adjusts the real-time states of angles in line with sensor signals to approach the reference/desired joint angles, which are the angles the servos must rotate to move the wrist point to an objective of known 3D coordinates, solved via inverse kinematics. In the mean time, fuzzy control also allows for speed control through a weighted combination of controllers with different feedback rate/gain. The fuzzy control model is designed such that the real-time relative position of the wrist from the destination, obtained from forward kinematics, decides which one out of the three subsets in the position fuzzy set, 'far', 'median range', and 'close', it falls into, which leads to a consequent subset in the speed fuzzy set, 'fast', 'median', and 'slow', that ultimately determines its speed.

The fuzzy models based on both types of the sensors, a gyro and an accelerometer, have been developed. The simplified fuzzy control model returns matching results to the design objective – the closer the wrist is to the target, the more the servo slows down, resulting in precise actuation at the cost of longer time/more steps. The manually-implemented model returns results with significant fluctuation while the model using the Matlab Fuzzy Logic toolbox returns results with less stiff change in speed. Fluctuation is a sign of fuzziness, and smoother change in speed is desired. The outcomes of both the later two models are valuable, but slightly miss the design objective. Further tuning of their parameters is required, and it is helpful to run all the models on the real robot to observe their real effects for evaluation.

Introduction

There is a dramatic projected growth of robotics in the coming future [1]. With growing labor cost in many developed and emerging (developed developing) economies around the world with China in particular [2], utilization of robots seems to be the only viable solution to maintain profitability and competiveness of the manufacturing industry. Robots are capable of performing high-precision tasks of uniform quality at a speed humans can hardly attain [3], reducing significant cost via shortening working hours from 8 down to 2 hours and lowering the expense on heat and lighting supply, etc [4]. As a consequence, industry begins to adopt robotics and replace human labor with robots.

There have been hundreds and thousands of differing robotic applications already. Among the leading commercial robots the most successful ‘humanoid’ android is the Rethink Robot ‘Baxter’ built by MIT [24], which is priced far below those of similar functionality. ‘Baxter’ is able to rethink as it remembers the trajectory of motion when its user moves its arms, and manages to repeat the same motion at any speed set by its user, greatly alleviating the heavy burden of tedious labor jobs.

The Humanoid Android Capstone Project aims at a robot consisting of multiple sub-systems, each of which outshines ‘Baxter’ and other similar products. The Humanoid Android possesses an anthropomorphic hand, i.e., a hand with a least four fingers, that enables minimum twenty degrees of freedom. It can perform a much broader scope of tasks than the two grippers ‘Baxter’ has, which only suit simple jobs like grabbing. Compared with existing fingered robotic hands, the Humanoid Android’s is designed to be light-weighted and built from cost-effective materials, a big breakthrough in robotics. While ‘Baxter’ sits still on its wheels, Humanoid Android will be standing on its two legs and walking like humans, which no commercialized robots have been capable of. Furthermore, Humanoid Android has advanced artificial intelligence embedded in its control. It realizes adaptive learning whereas ‘Baxter’ depends on manually guided learning. Overall, the project of Humanoid Android is a collection of technological breakthroughs in various areas of robotics.

Control is the core of a robot that executes all aspects of its performance. The humanoid Android runs on a model of fuzzy control at the request of the industrial sponsor, Bay Area IP. This paper focuses on the development of the arm control algorithm along with the implementation of the fuzzy logic. The objective of the control is to converge the wrist of the arm to a target location given the 3D coordinates of the destination.

Literature Review

Control

Robotics control has been evolving over the last half century, from ‘a mechanism with every joint controlled independently as a single-input/single-output linear system’ back in the early years to ‘interconnected nonlinear dynamics’ to ‘advanced control integrated with force and vision systems’; the complexity of tasks levels up over time accordingly, from simple tasks like material handling, spot welding and painting, to manipulator control and to mobile robots [17]. One state-of-arts humanoid robot is Honda’s Asimo Humanoid, so-called the world’s most advanced humanoid; its control successfully takes three difficult challenges into account – hybrid non-linear dynamics, unilateral constraints, and under-actuation, and realizes bipedal locomotion [17] [18].

Fuzzy logic

The term ‘fuzzy logic’ emerged during the development of the theory on fuzzy sets by Professor Lotfi Zadeh at University of California at Berkeley in 1965 [5] [6] [8]. The idea of fuzzy subset is characterized by assigning the respective degree of membership/truth to each element in a big set, so each subset becomes a vague proposition, i.e., ‘absolutely true’, ‘absolutely false’, or some intermediate truth degree [5] [8]. This set theory was not applied to control systems until the 70’s due to insufficiency in computing power prior to that time [6]. Professor Zadeh argued that feedback controllers could be programmed to accept imprecise input mixed with noise, which was apparently far easier to implement and could perhaps function more effectively like humans lack precise numerical information input and yet are capable of highly adaptive control [6]. The advantage of fuzzy logic lies in the simplicity of its design that the system will probably meet the task the first time even with no tweaking [6] or any prerequisite knowledge and understanding of control [11]. A fuzzy logic algorithm will take the form of a list of if/then statements when it comes to computing [5] [12].

Fuzzy logic has already become a standard technology and found its way to broad application all over the world [16]. Japan was the first country to embrace this technology and was aggressively building practical products around it, and then Europe, and lastly the United States [13] [14] [15] [16]. An interesting fuzzy product in people’s everyday lives is Zojirushi's NS-ZCC18 Neuro Fuzzy 10-cup version Rice Cooker, which switches cooking modes with the help of fuzzy logic [12].

In the document by Bonham-Carter, Graeme F. includes the algebraic operations of fuzzy logic and some very illustrative examples [9]. In the following example, each fuzzy set is assigned with one membership function on a continuous scale from 0 to 1 [8] [9]; the plot of the membership function is shown in Figure 1.

$$\mu(x) = \begin{cases} 0, & x \le 50 \\ \dfrac{x-50}{200}, & 50 < x \le 250 \\ 1, & x > 250 \end{cases}$$

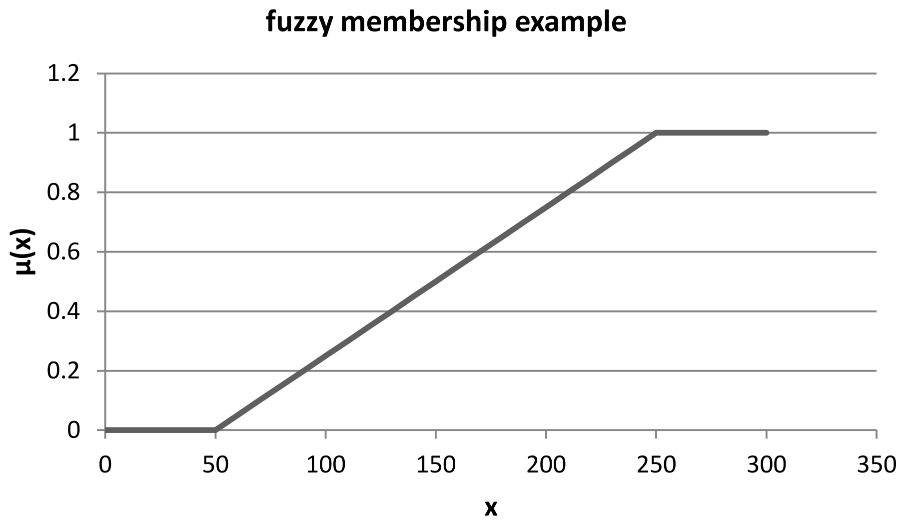

Figure 1 a fuzzy membership function plot [9]

In fuzzy control, the ultimate output is an aggregate output of differing independent fuzzy controllers. The membership function associated with a fuzzy set determines its influence, i.e., the weighting factor [7] of the controller belonging to that fuzzy set. Hence, depending on the set of 'vague proposition' one condition falls into, multiple controllers will contribute unevenly to the output.

The validity of fuzzy logic in conjunction with conventional control theorems has been widely researched. For instance, Lam, H. K. and Narimani, M. prove the stability of fuzzy control in their paper [10].

The fuzzy model is built on the universal feedback control form: $\dot{x} = Ax + Bu$ [10]

A feedback control is assigned with a weight, which is normalized grade of its respective fuzzy membership function: $w_i(x) \propto \mu_i(x)$, $\sum_{i=1}^{p} w_i(x) = 1$, $w_i(x) \in [0, 1]$ [10]

Also, let *u(t)* be a fuzzy controller, then same rules of fuzzy logic apply
$u(t) = \sum_{j=1}^{p} m_j G_j x(t)$, $\sum_{j=1}^{p} m_j(x(t)) = 1$, $m_j(x) \in [0, 1]$ [10]

So $\dot{x}(t) = \sum_{i=1}^{p} \sum_{j=1}^{p} w_i m_j (A_i + BG)\, x(t)$ [10]

Stability investigation of the continuous time feedback control system with Lyapunov function – $V(t) = x(t)^T P x(t)$ [10]

$\therefore \quad \dot{V}(t) = w_i m_j x(t)^T \left(\left(A_i + B_i G_j\right)^T P + P\left(A_i + B_i G_j\right) \right) x(t)$ [10]

$$\dot{V}(t) = \sum_{i=1}^{p}\sum_{j=1}^{p} w_i(m_j + \rho_j w_j - \rho_j w_j)z(t)^T \left(X(A_i + B_i G_j)^T + (A_i + B_i G_j)X\right) z(t)$$
$$+ w_i(w_j - \rho_j w_j)z(t)^T \Lambda_i z(t) - w_i(m_j - \rho_j w_j)z(t)^T \Lambda_i z(t)$$
$$+ w_i \rho_j z(t)^T (V_{ij} - V_{ij})z(t)$$

$$w_i w_j \rho_j z(t)^T \left(X(A_i + B_i G_j)^T + (A_i + B_i G_j)X - \Lambda_i - V_{ij}\right) z(t)$$
$$= \sum_{i=1}^{p}\sum_{j=1}^{p} + w_i(m_j - \rho_j w_j)z(t)^T \left(X(A_i + B_i G_j)^T + (A_i + B_i G_j)X - \Lambda_i\right) z(t) \quad [10]$$
$$+ w_i w_j z(t)^T (\Lambda_i + \rho_j V_{ij})z(t)$$

By letting $m_j - \rho_j w_j > 0$ and $X(A_i + B_i G_j)^T + (A_i + B_i G_j)X - \Lambda_i < 0 \quad for\ all\ i, j$

$$\dot{V}(t) \leq \sum_{i=1}^{p}\sum_{j=1}^{p} w_i w_j \rho_j z(t)^T \left(X(A_i + B_i G_j)^T + (A_i + B_i G_j)X - \Lambda_i - V_{ij}\right) z(t)$$
$$+ w_i w_j z(t)^T (\Lambda_i + \rho_j V_{ij})z(t)$$

$$= \sum_{i=1}^{p}\sum_{j=1}^{p} w_i z(t)^T R w_j z(t) + w_i z(t)^T S w_j z(t) \quad [10]$$

So the LMI-based stability is satisfied, i.e., R < 0, S < 0, and X > 0, as long as $m_j - \rho_j w_j > 0$ and *X, R, S, V, Λ* equal to their corresponding transpose matrices [10].

For learning purpose, the idea of fuzzy logic is modeled on top of the well-known control simulation, 'inverted pendulum on a cart' [20]. The algorithms are developed and simulated in Matlab. The switch of controllers is done via the Matlab built-in fuzzy-logic toolbox that generates a fuzzy membership function that manipulates the effect of the controller employed [21].

Methodology

Robotic arm design and assembling

The robotic arm is designed to resemble a human arm. Three servos are selected to constitute the shoulder, enabling rotational movements about each of the x, y and z axes of the three dimensional space, i.e., the arm opening/closing, moving forward/backward horizontally, and rotating vertically. The shoulder then connects to the upper arm, a steel rod, joining the forearm, a hollow rectangular steel tube, via another servo functioning as the elbow. This arm can reproduce any human arm motion until wrist.

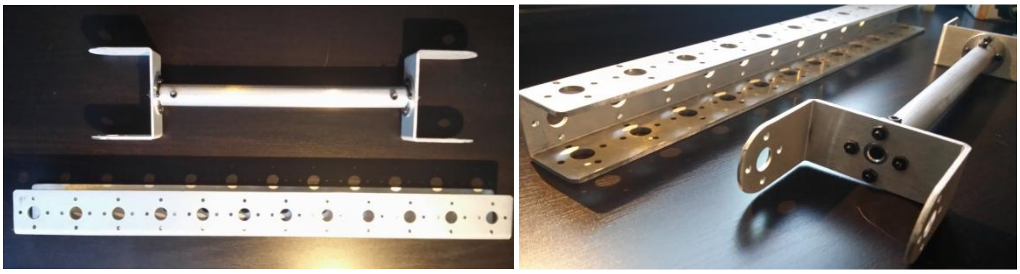

Figure 2A/B Robotic forearm and back arm

Figure 3A/B Servo HSR-5498SG and its holder

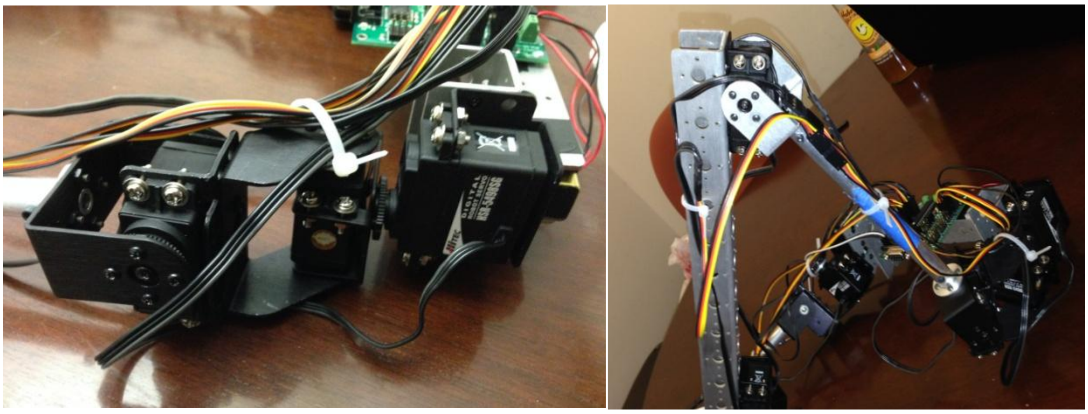

Figure 4A/B Assembled shoulder and elbow

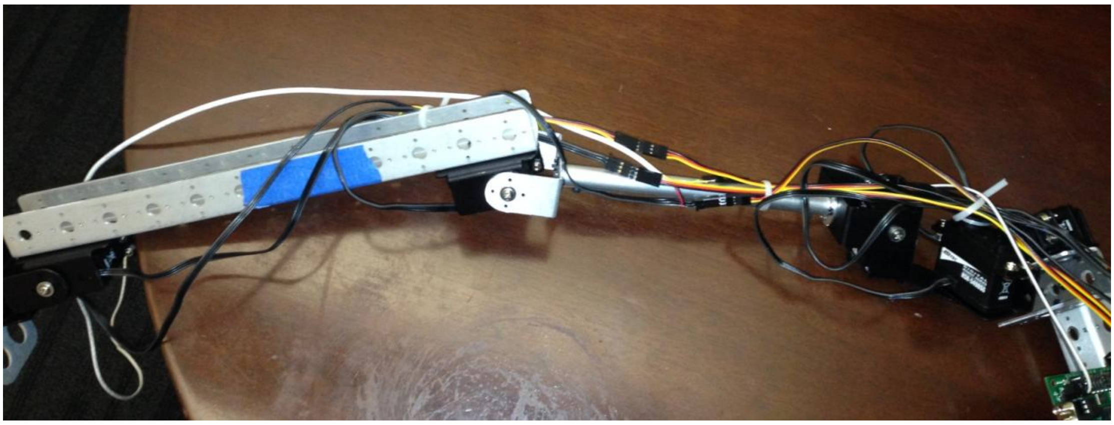

Figure 5 Assembled fully-open arm

Control development

A basic control algorithm ensures a generated signal input can be converted to physical movement in the real world via consistently adjusting in accordance with the sensor signals. Since the goal is to move the wrist to a destination given its 3D coordinates as inputs, inverse kinematics of Euclidean motion is used to solve for the angle of each joint. The resultant angles will be set as the desired outputs for the series of four servos.

Likewise, the real-time 3D location of the wrist, as well as the updated coordinate system, is obtained from a forward kinematics analysis. See the following sections for details.

The controller is designed to be a feedback loop that continuously adjusts the current state of the joint angles with respect to the inputs read from the sensors to approach the desired angle levels.

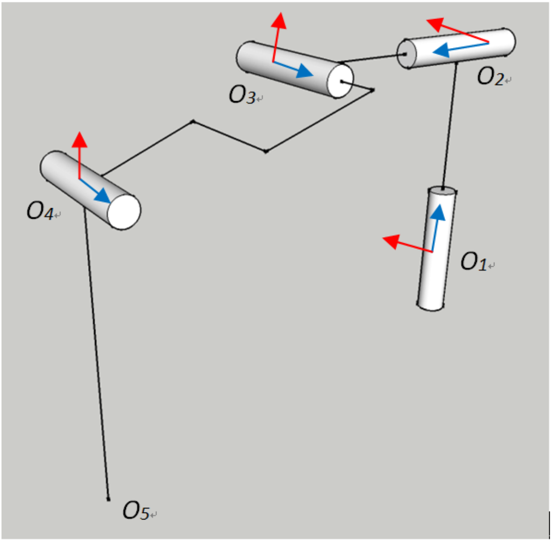

Figure 6 Arm manipulator

Forward kinematics

${}^{i\text{-}1}T_i$ is a homogeneous transformation that represents the coordinate system recursion over the space of joints vectors $[q_1\ q_2\ \ldots\ q_n]^T$ [21]

$$ {}^{i\text{-}1}T_i = \underset{\text{angle}}{\begin{bmatrix} e^{\theta_i k\times} & 0 \\ 0^T & 1 \end{bmatrix}} \underset{\text{offset}}{\begin{bmatrix} I & d_i k \\ 0^T & 1 \end{bmatrix}} \underset{\text{length}}{\begin{bmatrix} I & a_i i \\ 0^T & 1 \end{bmatrix}} \underset{\text{twist}}{\begin{bmatrix} e^{\alpha_i i\times} & 0 \\ 0^T & 1 \end{bmatrix}} = \begin{bmatrix} e^{\theta_i k\times} e^{\alpha_i i\times} & e^{\theta_i k\times} e^{\alpha_i i\times}(d_i k + a_i i) \\ 0^T & 1 \end{bmatrix} $$

1. angle: rotation of θ_i about k_{i-1}
2. offset: translation of di along k_{i-1}
3. length: translation of a_i along i_i, the common normal to joint axes i and i-1
4. twist: rotation of α_i about i_i

See Figure 6 for reference, *k-axes* are blue and *i-axes* are red.

By convention,

$$ {}^{i\text{-}1}T_i = \begin{bmatrix} e^{\theta_i k\times} R_i & e^{\theta_i k\times} \delta_i \\ 0^T & 1 \end{bmatrix} \quad \text{if joint } i \text{ is resolute} \quad [21] $$

$$ {}^{i\text{-}1}T_i = \begin{bmatrix} R_i & \delta_i + d_i k \\ 0^T & 1 \end{bmatrix} \quad \text{if joint } i \text{ is prismatic} \quad [21] $$

Therefore,

$$ \begin{bmatrix} C_n(q) & o_n(q) \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} C_0 & o_0 \\ 0^T & 1 \end{bmatrix} {}^0T_1(q_1)\, {}^1T_2(q_2) \ldots {}^{n\text{-}1}T_n(q_n) = \begin{bmatrix} C_0 & o_0 \\ 0^T & 1 \end{bmatrix} {}^0T_n(q_n) $$

C_i: the *ith* coordinate system

O_i: the joint location of the C_i

In the other words,

$$ C_i = C_{i-1}\, {}^{i\text{-}1}C_i = C_{i-1}\ e^{\theta_i k\times} e^{\alpha_i i\times}, \quad O_i = O_{i-1} + C_{i-1} e^{\theta_i k\times} e^{\alpha_i i\times}(d_i k + a_i i) $$

Following the Denavit-Hartenberg Convention [21], the results are listed in Table 1.

Note transformation starts at O_1 because the origin O_0 is known and there is no rotation or twist from O_0 to O_1 besides a fixed offset displacement. Please refer to Figure 6.

Link	θ_i	d_i	a_i	α_i
$O_{1\text{-}2}$	Θ_1	l	0	-90˚
$O_{2\text{-}3}$	Θ_2-90˚	l	0	90˚
$O_{3\text{-}4}$	Θ_3	0	L_a	0
$O_{4\text{-}5}$	Θ_4	0	L_b	0

$L_{a|b}$ is the length of the back arm and forearm; l is the offset from the center of one servo along its *k-axis* to the next.

Table 1 DH Convention Results

Inverse kinematics

Given the 3D coordinates of the target, O_5, imported from the computer vision subsystem; known O_1 *and the initial directional unit vector* k_1, (please refer to Figure 6)

If O_3 can be detected by a sensor, then

$\theta_1 = -\arcsin\left(\frac{[O_3 - O_1]\cdot\hat{k}}{[O_3 - O_1]\cdot\hat{\imath}}\right)$, displacement from joint 1 to 3, projected to *k-axis,* over that projected to *i-axis*; the projection is done by dot product with the corresponding unit axis;

$C_3k_3 = C_1^1C_2^2C_3k_3 = C_1k_1 \rightarrow k_3\ , \theta_2 = arcos\left(\frac{dot(k_1,k_3)}{|k_1||k_3|}\right)$, the angle between the reference *k-axis* and its transformed unit vector;

$\theta_3 = arcos\left(\frac{l^2 + L^2 - norm(O_4 - O_2, 2)^2}{2lL}\right)$, Cosine Law;

$\theta_4 = arcos\left(\frac{L^2 + L^2 - norm(O_5 - O_3, 2)^2}{2L^2}\right)$, Cosine Law plus a tweak

If O_3 cannot be detected by a sensor, then predict it by trying all possible angles for θ_1, θ_2 ~ [0, 180°], range of the servos, with one step of 0.5°, and find all combinations of the angles such that $O_5 - O_3 \perp k_3$; out of all the possible angle sets, select the set with the minimum sum so the two servos will turn the least amount, leading to the most efficient arm movement. Note O_3 *and* k_3 are obtained using forward kinematics.

Fuzzy logic development

Fuzzy logic is implemented on top of a conventional arm control algorithm. Switch among the controllers in different situations is achieved by changing one key parameter of a single controller, e.g., gain, according to the context. In other words, one single controller behaves as completely different controllers following the change of one key parameter. The key parameter in this project is chosen to be the angular speed of the servos while the fuzzy logic condition is the relative position of the wrist with respect to the target position.

The real-time wrist position determines which controller to be used. The fuzzy logic is designed that the closer the wrist is to the target position, the lower the servo speed drops because of a common trade-off between speed and accuracy/precision. While the wrist position is far away from the target, it is perfectly logical to move at a high speed as big steps bring efficiency though leaving a coarse trajectory. However, as the wrist position gets closer to the target, the servo slows down, and it takes longer or more steps to reach the objective angle, resulting in more frequent adjustments and a finer trajectory.

Microprocessor code

Figure 7 Microprocessor PIC 32M

Finally, the fuzzy logic algorithm in Matlab code will be translated into C so that it can be loaded into the microprocessor where servo wires can be directly plugged in to actually move the robotic arm. Details of this fuzzy logic-based control, i.e., the exact optimal number of controllers being employed and the optimal value of the key parameter – angular speed – for each controller is yet to be determined via tuning the parameters in a number of trials. The trial-and-error method is completely based on qualitative results observed and recorded.

Discussion

The idea of fuzzy logic is implemented in the servo speed control. Three rules of fuzzy logic are designed to be followed: if the real-time wrist point is far from the target, then the servos will rotate faster to reach the desired angles; if the wrist point is in a median range from the target, then the servos will move at a moderate angular speed; and lastly, if the target is close to the target, then the servos will slow down. The rules enable finer and more frequent servo actuation as the real-time angle of each servo approaches its desired level computed from the given 3D coordinates of a target via inverse kinematics. The fuzzy logic condition – the relative position of the wrist to the destination – is the fraction of the spherical range, on the boundary of which lies the real-time wrist point, out of the range of the destination. The real-time spherical range, i.e., the displacement from the origin, is obtained by converting the real-time servo angles to the real-time wrist location through forward kinematics. The fuzzy set of relative position, {'far', 'median range', 'close'}, is thus on the scale of *0:1*. The following designs show the simulated results of fuzzy control with variation in the way to divide by the boundaries of the position fuzzy subsets and their inference of the speed fuzzy set, {'slow', 'moderate', and 'fast'}. An illustration of the inference between the position and speed fuzzy sets is shown in Figure 8A/B/C. In the example, the position fuzzy set is modeled with three overlapping Gaussian distributions shown in Figure 8B while the speed fuzzy set is filled with three equal triangular distributions shown in Figure 8C.

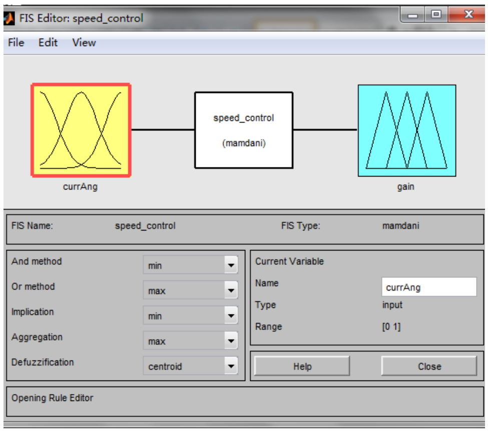

Figure 8A Matlab Fuzzy-logic toolbox (inference)

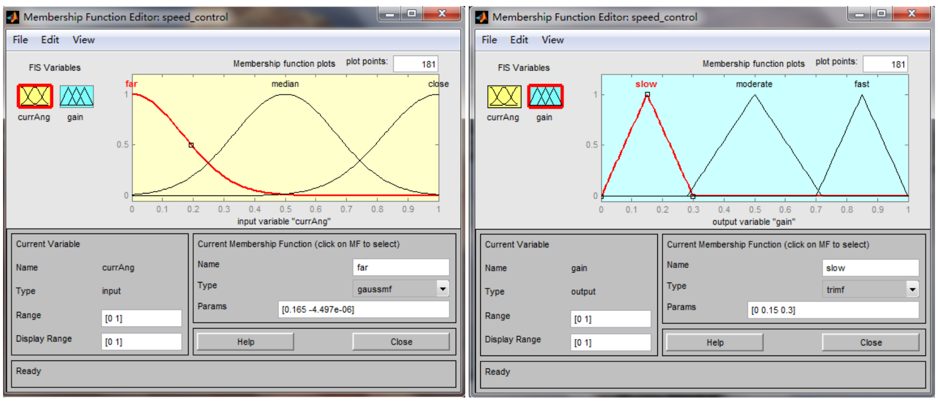

Figure 8B/C Matlab Fuzzy-logic toolbox (input and output membership funtions/distributions)

The output of the control system is a pulse that goes directly into a servo, and its pulse width is proportional to an angle a servo will rotate [23]. Therefore, a pulse illustrates an angular speed since its wavelength equals to a multiple of an angle and its integral over time must be a multiple of the real-time angle a servo rotates. A wave generator is used in Simulink to produce a square wave of a specified wavelength with all the negative values zerod. The speed fuzzy set determines the gain of the feedback loop in the controller, and consequently which speed controller in the fuzzy model predominates. See Figure 10, 15, 17, 23 for various designs of the fuzzy control model. Figure 10, 17 show the designs without using the fuzzy logic toolbox that comes with Matlab; Figure 15, 23 show that with Matlab Fuzzy Logic Toolbox.

The sensor in Figure 9 possesses both a 3-axis gyro and a 3-axis accelerometer. Either provides sufficient information. A gyro measures angular speed. An accelerometer detects acceleration in a 3D space, from which together with the specific geometry of the mechanical design, angular acceleration can be found.

Figure 9 Sensor MPU-6050

For both the gyro-based and accelerometer-based designs, the resultant wave signals that are the controller outputs that would be sent to control the servos in multiple designs are recorded –

1. Non-fuzzy control
    a) See Figure 11 for gyro-based non-fuzzy design
    b) See Figure 18 for accelerometer-based non-fuzzy design
2. Simplified fuzzy control – the scale of relative position fuzzy set is uniformly divided into three, and each follows the three rules to match one subset in the speed fuzzy set, but each speed subset contains one fixed gain, i.e., ‘fast’ = 10, ‘moderate’=3, ‘slow’ = 0.8.
    a) See Figure 12, 13 for the simplified version based on gyro
    b) See Figure 19, 20 for the simplified version based on accelerometer
3. A manually implemented fuzzy control – each subset in the relative position fuzzy set follows a Gaussian distribution with a mean of 0, 0.5, 1, respectively, and a standard deviation of 5/3, e.g., shown in Figure 8B; each Gaussian function determines the weight of the corresponding speed subset following the three rules, and each speed subset contains one fixed gain like that in the previous bullet
    a) See Figure 14 for the manually implemented design based on gyro
    b) See Figure 21, 22 for the manually implemented design based on accelerometer
4. Fuzzy control using the Matlab fuzzy-logic toolbox
    a) See Figure 16 for the design based on gyro using the toolbox
    b) See Figure 24 for the design based on accelerometer using the toolbox

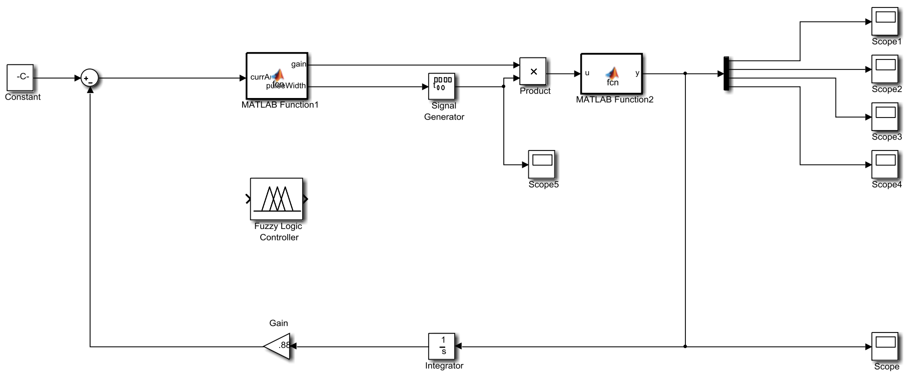

Figure 10 gyro-based fuzzy control

Figure 11 gyro-based non-fuzzy control

Figure 12 gyro-based simplified fuzzy logic I

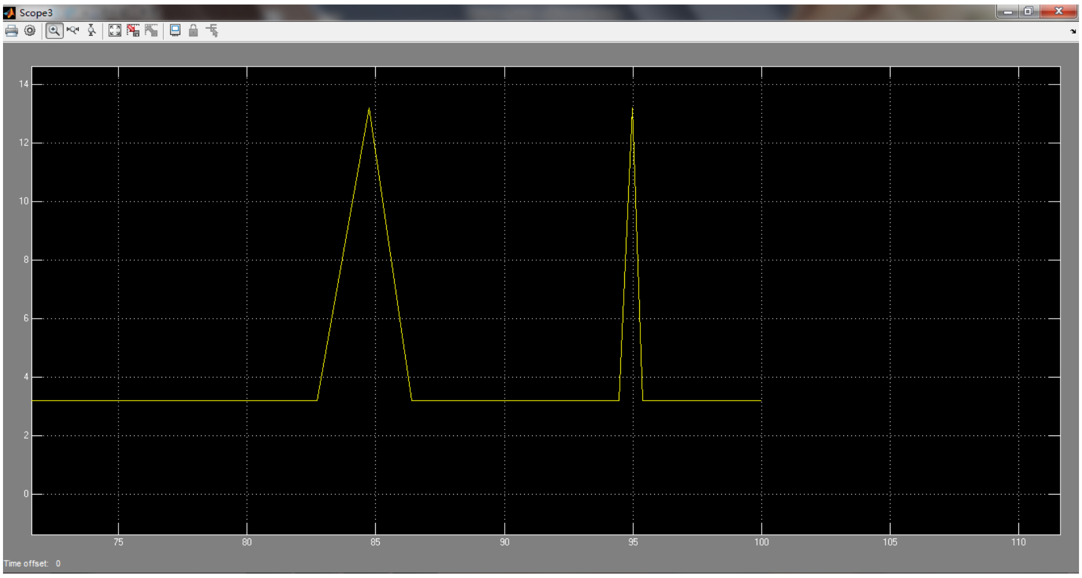

Figure 13 gyro-based simplified fuzzy logic II

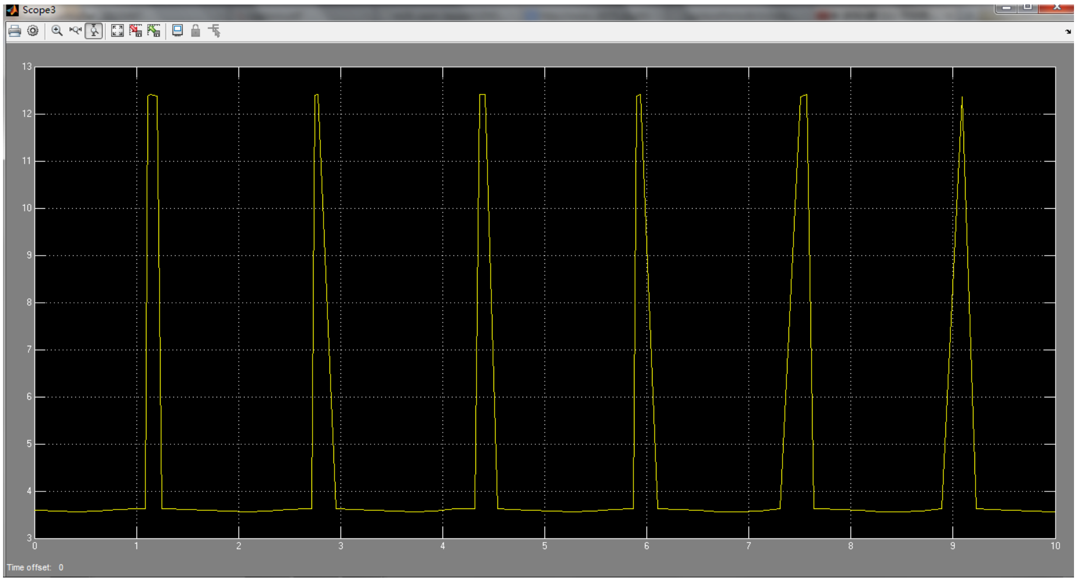

Figure 14 gyro-based manually-implemented fuzzy logic

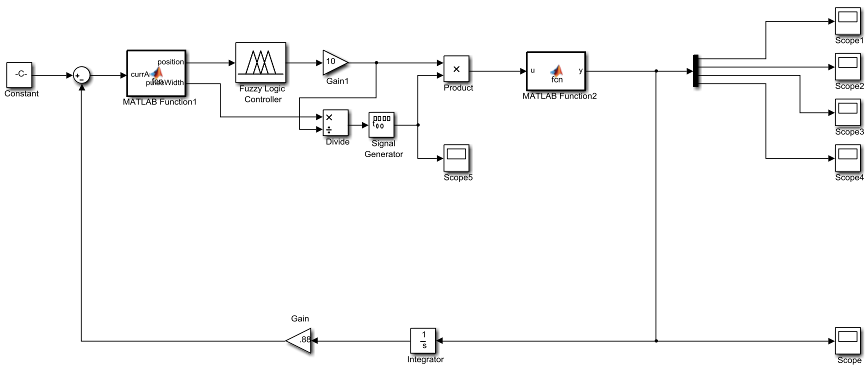

Figure 15 gyro-based fuzzy control with Matlab Fuzzy Logic Toolbox

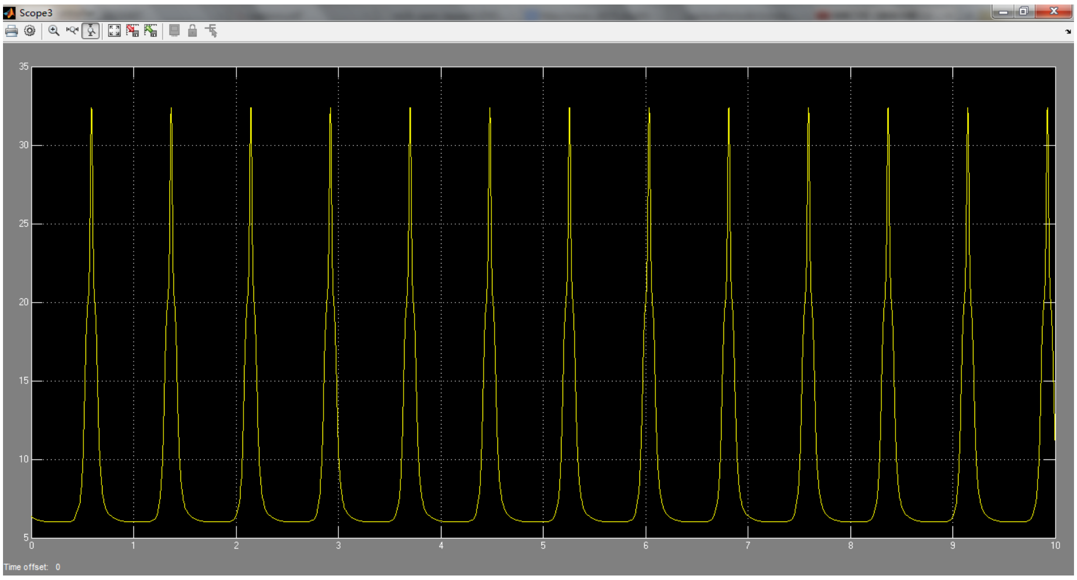

Figure 16 gyro-based Matlab Fuzzy Logic Toolbox

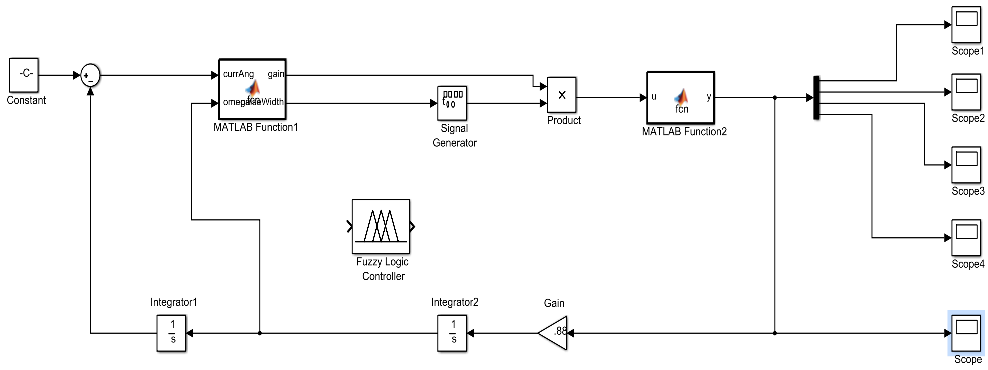

Figure 17 accelerometer-based fuzzy control

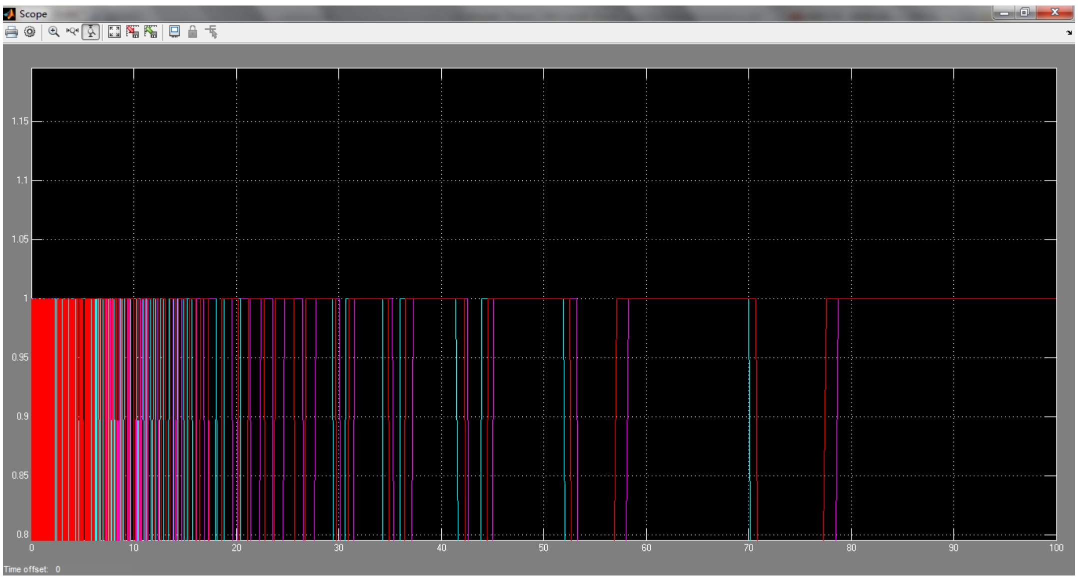

Figure 18 accelerometer-based non-fuzzy control

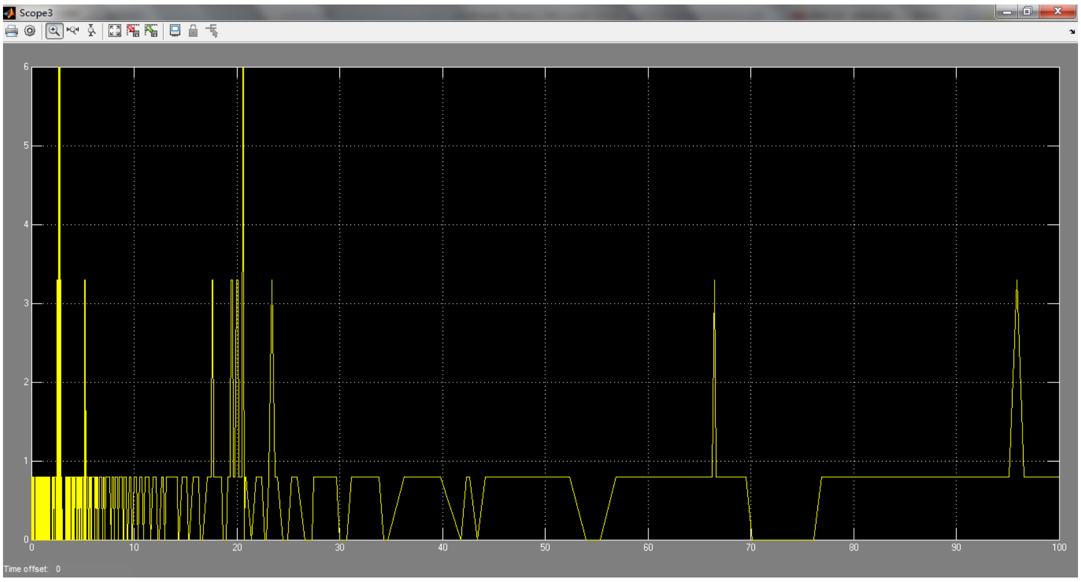

Figure 19 accelerometer-based simplified fuzzy logic (single angle)

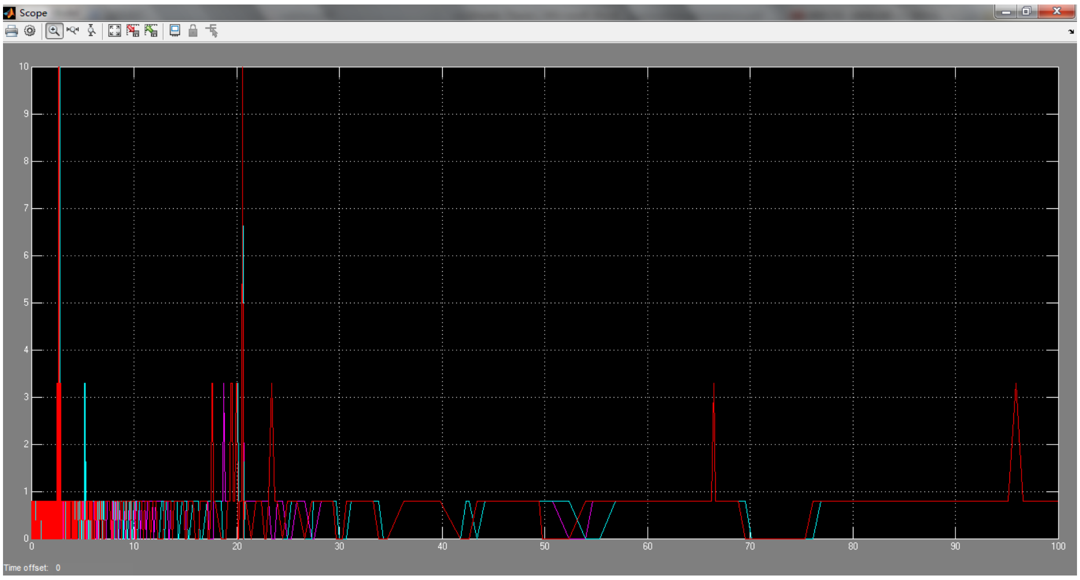

Figure 20 accelerometer-based simplified fuzzy logic (all angles)

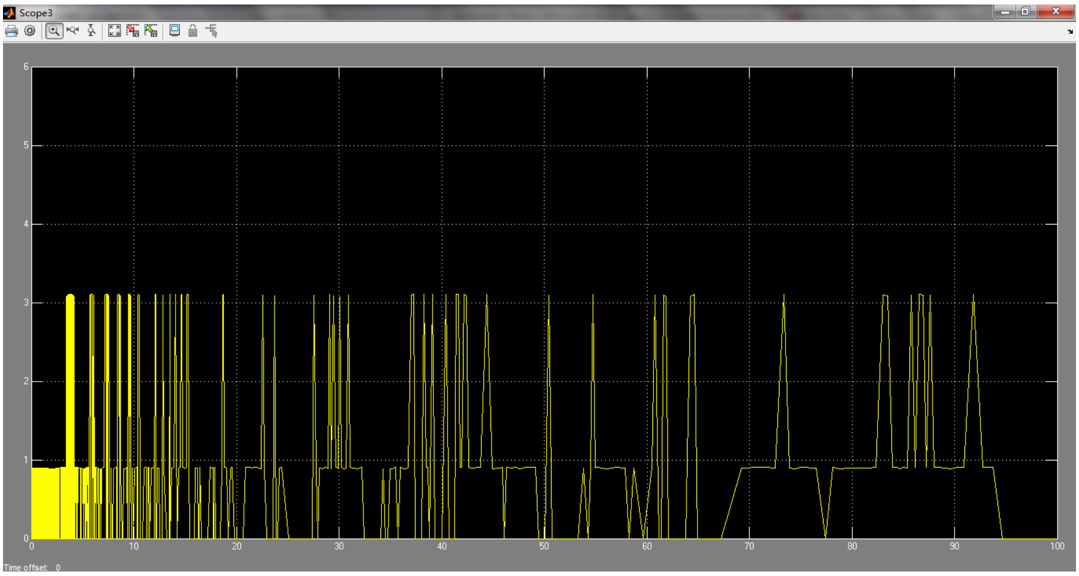

Figure 21 accelerometer-based manually-implemented fuzzy logic (single angle)

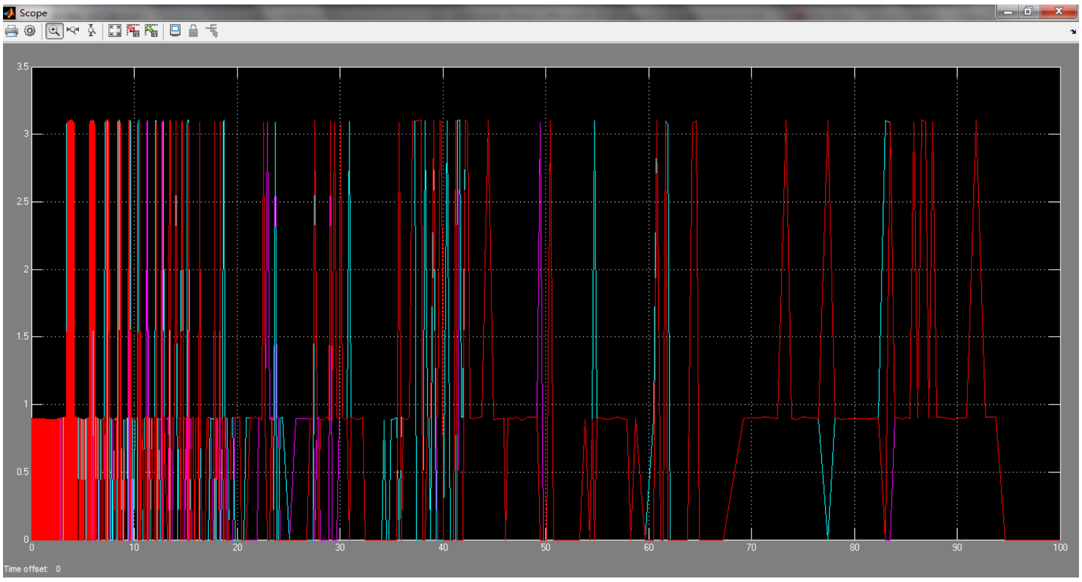

Figure 22 accelerometer-based manually-implemented fuzzy logic (all angles)

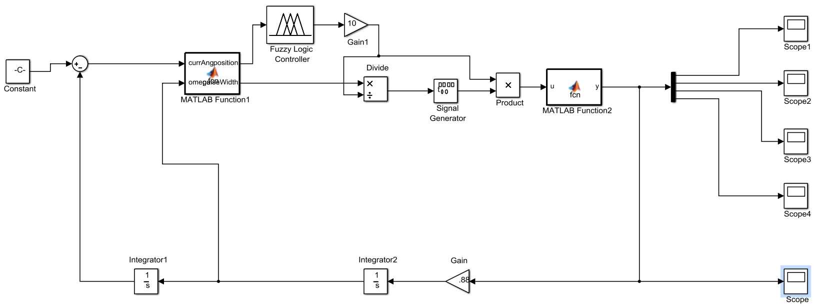

Figure 23 accelerometer-based fuzzy control with the Matlab Fuzzy Logic Toolbox

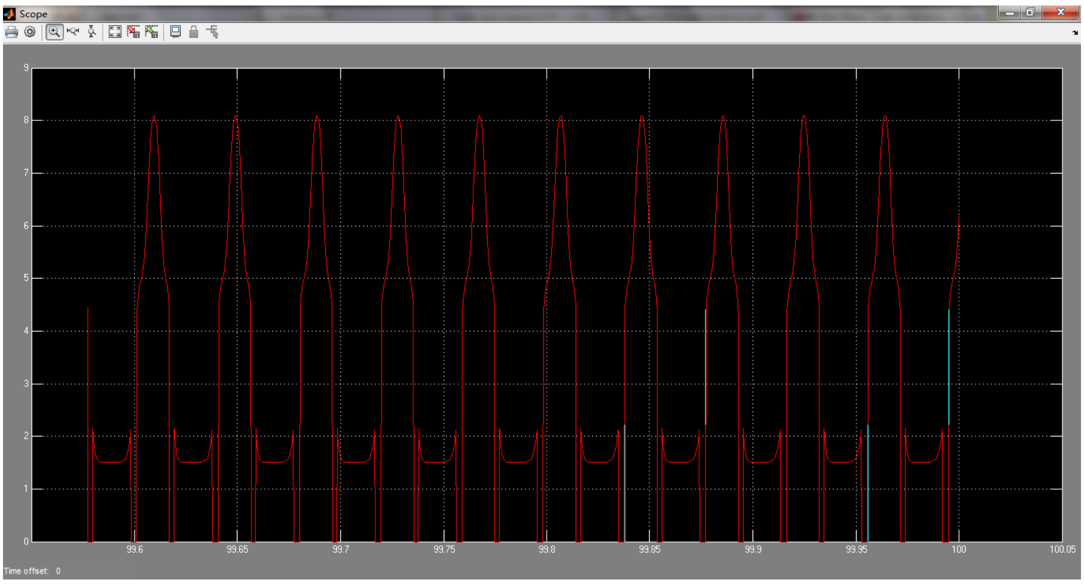

Figure 24 accelerometer-based fuzzy logic toolbox

The results clearly demonstrate that incorporation of fuzzy logic in the design enables speed changes that appear as spikes in the plots. In comparison to the non-fuzzy control shown in Figure 11, 18, the simplified fuzzy control leads to expected changes in speed, i.e., the change in amplitude directly reflects the gain or the new speed updated in the controller. The trend of the speed lowers with time shown in Figure 12, 13, 19, 20, matching the design intent that the servos rotate at a fast speed when the wrist point is far from the target but at a slow speed as the wrist point gets close to the destination. The manually-implemented fuzzy logic adds fluctuation to the pulse waves making the speed control fuzzier shown in Figure 14, 21, 22. The results from the simulations including Matlab Fuzzy Logic Toolbox are a bit difficult to interpret, but they do show less stiff speed change shown in Figure 16, 24, which is desirable. In the manually-implemented and the Matlab toolbox-based fuzzy controls, the spikes do not decrease with time but repeat the same patterns of speed change periodically, which does not coincide with the original design intent. That means some parameters, i.e., the weight assigned to each element of the speed fuzzy set, must be tuned further to produce an ideal signal. It is highly recommended to tune the parameters in the actual robot because there is one chip responsible for automatically generating a pulse wave to control a servo, and the microprocessor only needs to be programmed to compute the input angle and angular speed that will be passed to that chip. Furthermore, the sensor readings will be accurate and more relevant reflections of the combined friction loss in servo actuation, whereas in Matlab the sensor reading is assumed to be a simple linear multiple of the output.

Conclusion

Fuzzy logic triggers a decision making process based on qualitative classification of quantitative evidence. This idea can merge with traditional control, leading to an artificially intelligent control system. Fuzzy control enables two or more different controllers to influence the physical performance unequally according to the situation. It has become a viable technology in industry and finds its ways into many commercial products, and in academia, its compatibility with established control theories has been studied and proven.

In this capstone project, Humanoid Android Subsystems, a fuzzy control algorithm for the robotic arm has been developed. It allows speed change of the joint servos, which should enable more precise target location and more efficient arm movement. The simplified fuzzy control model involves switching among controllers, and the effect of fuzzy logic is evident as its simulated signal shows shifts in differing controllers' influence over time, which meets the expectation. This model is the most suitable one to be directly re-used in the real robot due to its simplicity and apparent performance. The other two fuzzy control models involve Gaussian distributions when it comes to classification within the fuzzy set; the overlapping between the Gaussian distributions definitely makes the classification and the follow-on inference fuzzier, but unfortunately, the original design intent is not clearly inferable from the result. Based on the simulated results, their parameters need to be further tuned though they provide useful information. Although there are so many uncertainties associated with selecting the parameters, such as the number of different controllers in one fuzzy set, the types of distributions and respective weight assigned to influence the outcome, and how the output fuzzy set is inferred from the input, etc, an algorithm based on fuzzy logic is likely to work on the first trial prior to any tweaking. So it is highly recommended to try running the designs on the assembled robot and observe its qualitative performance, which will be a good start point for tuning.

Reference

1. Hanlon, M. (2008, December). Personal Robotics Industry set for massive growth. Retrieved from http://www.gizmag.com/go/2407/
2. United States Department of Commerce (2010). ASSESS COSTS EVERYWHERE. Retrieved from http://acetool.commerce.gov/labor-costs
3. Max, E. (2013, November). Advanced Robotics Boost the Manufacturing Sector. Retrieved from http://blog.max.exactamerica.com/manufacturing-insights/advanced-robotics-in-manufacturing
4. ABB Group (2009, June). 10 Good Reasons to Invest in Robots. Retrieved from http://www04.abb.com/global/seitp/seitp202.nsf/c71c66c1f02e6575c125711f004660e6/89654de35f1b9214c12575d300435264/$FILE/10+good_reasons_June_+09.pdf
5. Stanford Encyclopedia of Philosophy (2010, August). Retrieved from http://plato.stanford.edu/entries/logic-fuzzy/#3
6. Yager, et al. (1987). Fuzzy Sets and Applications: Selected Papers by L.A. Zadeh. *John Wiley, New York.*
7. Kaehler, S. D. FUZZY LOGIC – AN INTRODUCTION. Retrieved from http://www.seattlerobotics.org/encoder/mar98/fuz/fl_part4.html
8. Zadeh L. A. (1965). Fuzzy Sets. *Information and Control, 8(3), pp. 338-353.*
9. Bonham-Carter, Graeme F. Fuzzy Logic Method. *Geographic Information Systems for Geoscientists, Modelling with GIS, Chapter 9, Fuzzy Logic section with related tables and figures.*
10. Lam, H. K. and Narimani, M. (2009, August). Stability Analysis and Performance Design for Fuzzy-Model-Based Control System Under Imperfect Premise Matching. *IEEE TRANSACTIONS ON FUZZY SYSTEMS, VOL. 17, NO. 4.*
11. Society of Robots (2014). PROGRAMMING – FUZZY LOGIC. Retrieved from http://www.societyofrobots.com/programming_fuzzy_logic.shtml
12. Toothman, J. (2014). How Rice Cookers Work. Retrieved from http://home.howstuffworks.com/rice-cooker2.htm
13. Smith E. T. (1993, February), Why the Japanese are Going in for this 'Fuzzy Logic', *Business Week.*
14. Europe Gets into Fuzzy Logic. (1991, November). *Electronics Engineering Times.*
15. U.S. Loses Focus on Fuzzy Logic. (1990, June). *Machine Design.*
16. fuzzyTECH (2000). Industrial application of Fuzzy Logic Control. *INFORM.* Retrieved from www.fuzzytech.com
17. Spong, M. W. & Fujita, M. (2011). Control in Robotics. *The Impact of Control Technology, T. Samad and A.M. Annaswamy eds.*
18. Honda (2014). ASIMO. Retrieved from www.asimo.honda.com
19. Filo, G. MODELLING OF FUZZY LOGIC CONTROL SYSTEM USING THE MATLAB SIMULINK PROGRAM.
20. MathWorks (2014). fuzzy-logic tutorial. Retrieved from http://www.mathworks.com/products/fuzzy-logic/videos.html
21. Salcudean, T. (2014). Forward Kinematics notes. Retrieved from

https://courses.ece.ubc.ca/487/priv/Notes2011/ch2_forkin.pdf

22. Salcudean, T. (2014). Forward Kinematics notes. Retrieved from https://courses.ece.ubc.ca/487/priv/Notes2011/ch4_invkin.pdf
23. Society of Robots (2014). ACTUATORS – SERVOS. Retrieved from http://www.societyofrobots.com/actuators_servos.shtml#current
24. Wang, J (2013). 8 Companies Leading the Charge for Commercial-Use Robotics. Retrieved from http://www.entrepreneur.com/article/226402#