Embedded Module Testing Robot using Neural Network

S. Thaminmun Ansari[1], S. Uma[2]
[1] ME (Embedded System Technologies), Electrical & Electronics Engineering Department,
[2] Assit. Prof. Electrical and Electronics Engineering Department,
Anna University, Chennai, India

***Abstract* — The purpose of this paper is a simple and low cost approach to design and implement a programmable robot which can be used to automate embedded module testing. User can simply train different embedded module features to be tested to robot and can use it for testing automation. The robot shall be commanded from PC to do different task using serial communication. This feature shall be very useful specifically for embedded auto part designing industries to meet desired quality, time line and resource requirement of testing.**

***Index Terms*— Testing automation using robot, Robotics in Industrial automation and control, Artificial Intelligence in Robotics, Artificial Intelligence in Industrial automation.**

I. INTRODUCTION

Today technology is growing up with very high speed. This growth affects every field. A decade ago meters in a car is used to indicate speed, RPM, fuel level and engine temperature only. But today advancement in technology expects the meter to show infotainment details, navigation maps, system configuration settings, terrain response, my key details, traffic recognition symbols, social network connectivity and telephone connectivity information etc. As evolution of networking technologies like Controller Area Network (CAN), Local Interconnect Network (LIN) & Media Oriented System Transport (MOST), many Electronic Control Units (ECU) in vehicle networks are interconnected and more sophisticated features are supported in a single ECU. This increases complexity of the product. As complexity of the product is increased, complexity of software designed for the product is also increased which in turn increases no of test cases, defect density, resource involved for testing and time etc. Meter of a car has key areas like Telltales (a glowing bitmap to annunciate fault), warning (Text display to denote fault), menu navigation, Chimes (audio beeping) etc. Testing of these feature consumes lot of time and resource.

Today most of the auto part manufacturing industries test these features manually using more number of human resources as testing involves verification of visual output like image or text. Some of them use high cost software like LABVIEW, MATLAB and LABCAR along with respective hardware interfaces. Some of the modules in Device Under Test (DUT) need button press to trigger output and some outputs audio signals. There are lot of researches already in place to recognize images and text for various application like face recognition, signature recognition and shape recognition.

After doing lot of analysis in all those research, Paper Device a customized technique to design a robot which can automate testing of these features using predefined test sequence using neural network. Upon triggering visual Feature input robot shall verify the output images and text without human intervention and button press input to ECU module is triggered by mechanical arm of the robot.

II. METHODOLOGY

Button matrix is derived from the layout of the button in the test box. Robotic arm with servo motor controlled by a microcontroller is trained to move required x and y position from the reference to reach intended button in the test box. The command to press required button and key held time is transmitted from Graphical User Interface (GUI) in the PC to controller using serial communication protocol. Upon receiving the required command robotic arm press the commanded button in the test box for key held time. Telltale, warning text and menu outputs are verified by capturing cluster display image using a webcam and recognizing them using neural network.

III. BUTTON PRESS SIMULATION

A. Hardware

A mechanical setup to mimic human hand is prepared using aluminum foil to trigger button press. It has three joints with certain gap. The gap between each joint is designed based on the max length to be reached by robotic arm to execute the button press. This is called as robotic work space.

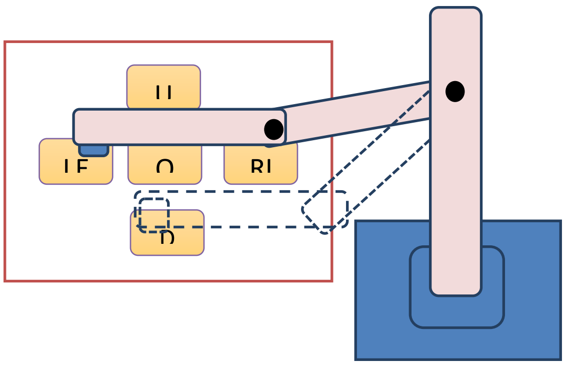

Fig 1: Robotic arm Mechanical Setup

B. *Servomotor*

Each joint is powered by servo motor for rotational and translation movement. Servo motors are special type of motor whose shaft shall be positioned to a specific angular position by sending coded signal. As long as coded signal exist on the control line, the servo will maintain the angular position of the shaft. If the coded signal changes, then angular position of the shaft changes. The angle is determined by the duration of a pulse that is applied to control wire. This is called pulse width modulation (PWM). The servo expects to see a pulse every 20ms. The length of the pulse will determine how far the motor turns. For example 1.5ms pulse will make the motor to 90 degree position.

C. *PWM Signal Generation*

In order to control each servo in robotic arm we need PWM pulse with 20ms periodicity and 1-2ms ON time. Hence PIC 16F877A controller is selected to generate the programmable PWM pulses. This controller is an 8 bit controller with 4 programmable IO port, on chip timer with Serial communication interface. As controller is operated with 12 Mhz frequency it is unable to use available timer to trigger configurable PWM in the controller. Hence every 0.1ms an interrupt is generated using Timer 0 module of the controller.
For 20 ms = 200 * 0.1 ms formula is used

D. *GUI design and communication*

In order to control ON and off time PWM signal Scilab based GUI is designed and control commands are transmitted through serial port using serial communication tool box. Scilab is Matlab like software and has image processing tool box, Serial communication tool box and GUI elements those are required feature for the project [10]. Initially X. Y position of each button and respective stepper motor to be driven is stored in the controller memory through learning mode. Scilab GUI transmit key needs to be pressed and key held time from PC. Software program flashed in controller monitors serial port for commands upon power up. Upon receiving the valid command from PC controller decodes it and identifies required button needs to be pressed. Then required servo motors are driven to achieve intended button press.

Button press automation shall be described using flow diagram in Fig 2

IV. TELLTALE & WARNING RECOGNISTION

A. *Thinning algorithm*

The telltale or warning output is captured using webcam and is transferred to Scilab image processing tool box. Images are converted in to binary images and then skeleton of the image with region R and border B is found using thinning algorithm. The algorithm consist of two successive passes of two basic steps applied to contour point of the given region where contour point is any pixel with value 1 and having at least 8 neighbor values 0. With reference to 8 neighborhood notation shown in fig 3 step 1 flags the contour point p1 for deletion if the following conditions are satisfied [16]

a) $2 \le N(p1) \le 6$ (1)

b) $T(p1) = 1$ (2)

c) $p2 * p4 * p6 = 0$ (3)

d) $p4 * p6 * p8 = 0$ (4)

When N(p1) is number of nonzero neighbors of p1; That is

$N(p1)=p2+p3+p4+p5+p6+p7+p8+p9$ (5)

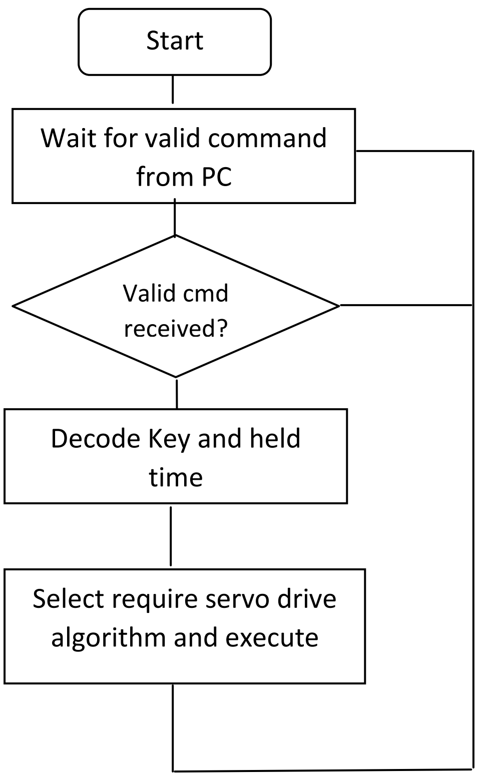
Start
Wait for valid command from PC
Valid cmd received?
Decode Key and held time
Select require servo drive algorithm and execute

Fig 2. Software flow for button press automation

$p9$	$p2$	$p3$
$p8$	$p1$	$p4$
$p7$	$p6$	$p5$

Fig 3: Eight neighborhood notation

And T(p1) is number of 0-1 transition in the ordered sequence p2,p3,…..p8, p9, p2

In step 2, the conditions (a) and (b) remains same, but condition (c) and (d) are changed to

(c') $p2 * p4 * p8 = 0$ (6)

(d') $p2 * p6 * p8 = 0$ (7)

Once the skeleton of the image is framed they recognized through pattern matching using neural network.

V. PATTEN MATHING USING NEURAL NETWORK

In order to recognize telltale and warning paper uses neural network. Neural network is connectionist computational system and does not follow linear path, rather information is processed collectively, in parallel throughout a network of nodes [15]. One of the key element of neural network is its ability to learn. A neural network is not just a complex system, but complex adaptive system, meaning it can change its internal structure based on the information flowing through it, typically, this is achieved through the adjusting of weights, a number that controls the signal between the two neurons, if the network generates correct output there is no need to adjust the weights. However, if the network generates poor output an error then system adapts, altering the weights in order to improve subsequent results.

There are several strategies for learning and paper uses supervised learning, essentially, a strategy that involves a teacher that is smarter than the network itself. In our case teacher shows the network a bunch of telltales, and the teacher already knows the name associated with each telltale. The network make its guesses, the teacher provides the network with the answers. The network can then compare its answers to the known "correct" ones and make adjustments according to its errors. The neural networks are complicated and difficult, they involves all sort of fancy mathematic

A. *The perceptron*

A perceptron is the simplest neural network possible a computational model of a single neuron. A perceptron consist of one are more inputs, a processor, and a single output. Perceptron follows the "feed- forward" model meaning inputs are sent into the neuron and processed and result in an output. Inputs come in, output goes out. Say we have a perception with two inputs let's call them x1 and x2

Input 0: X1 = 12
Input 1: X2= 4

Each input that is sent into the neuron must first be weighted. i.e multiplied by some value often a number between (-1 and 1). When creating a perception, we will typically begin by assigning random weight, here let's give the input the following weights:
Weight 0: 0.5
Weight 1: -1
We take each input and multiply it by its weight
Input 0 * weight 0 -> 12 * 0.5 =6 (8)
Input 1 * weight 1 -> 4 * -1 = -4 (9)
Weighted inputs are then summed
Sum= 6+ -4 = 2 (10)
The output of the perceptron is generated by passing that sum through an activation function in case of a simple binary output. The activation function is what tells the

perceptron whether to “fire” or not. You can envision an LED connected to the output signal: if it fires the light goes on: if not its stays off

Activation function can get a little bit hairy, if you start reading one of those artificial intelligence text books looking for more info about activation functions, you may soon find yourself reaching for a calculus textbook. However, with our friend the perceptron we are going to do something really easy, Let make the activation function the sign of the sum. In other words, if the sum is a positive number, the output is 1; if it is negative, the output is -1

Output =sing(sum) => sign (2) = +1 (11)

Let us review and condense there steps so we can implement them with the code snippet.

The perceptron algorithm:

1. For every input, multiply that input by its weight
2. Sum all of the weighted inputs
3. Compute the output of the perceptron based on that sum passed through an activation function (sign of sum)

B. The pattern matching

We are going to apply this algorithm to find right telltale after extracting the skeleton of the telltale in two dimensional space by computing whether points are outside or inside the reference points. The perceptron has two input x and y coordinate. Using a sign activation function, the output will either be -1 or 1 the input data is classified according to the sign of the output, in the above diagram, we can see how each pixel is either within or outside the edge. Presumably we could now create a perceptron object and ask it to make a guess for any given point

Did the perceptron get it right? At this point, the perceptron has no better than 50/50 chance of arriving at the right answer, Remember, when we created it, we gave each weight a random value. A neural network is not a magic. It is going to employ the method of supervised learning with this method the network is provided with inputs for which there is known answer. This way the network can find out if it has made a correct guess. If it is incorrect, the network can learn from its mistake and adjust its weights. The process is as follows

1. Provide the perceptron with input for which there is known answer
2. Ask the perceptron to guess an answer
3. Compute the error. (Did it get the right answer or wrong?
4. Adjust all the weights according to the error
5. Return to step 1 and repeat

Step one through 4 can be packed into a function. Before we can write the entire function. However, we need to examine step3 and 4 in more detail. How do we define the perceptron`s error? And how should we adjust the weights according to this error? The perception error can be defined as the difference between the desired answer and its guess.

ERROR= DESIRED OUTPUT- GUESS OUTPUT (12)

In the case of perception the output has only two possible values +1 or -1. This means there are only three possible errors. If the perceptron guess the correct answer, then the guess equals the desired output and the error is 0. If the correct answer is -1 and we have guessed +1 then the error is -2. If the correct answer is+1 and we have guessed -1 then the error is +2

Table 1: Desired Vs Guess

Desired	Guess	Error
-1	-1	0
-1	+1	-2
+1	-1	+2
+1	+1	0

The error is the determining factor in how the perceptron`s weights should be adjusted. For any given weight, what we are looking to calculate is the change in the weight. Often called delta weight

NEW WEIGHT = WEIGHT + DELTA WEIGHT (13)

DELTA WEIGHT= ERROR * INPUT (14)

Therefore

NEW WEIGHT = WEIGHT + ERROR * INPUT

Hence visual output like telltales and warning texts are recognized using neural network

ACKNOWLEDGMENT

We acknowledge all staff members of embedded system technology department for their timely support to complete this project

REFERENCES

[1] A. Rattarangsi and R. T. Chin, “Scale-based detection of corners of planar curves,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 14, no. 4, pp. 430–448, Apr 1992

[2] B. Kégl and A. Krzy˙zak, “Piecewise linear skeletonization using prin- cipal curves,” IEEE Trans.

Pattern Anal. Mach. Intell., vol. 24, no. 1, pp. 59–74, Jan. 2002

[3] C. Wang, D. J. Cannon, R. T. Kumara, and G. Lu, "A skeleton and neural network-based approach for identifying cosmetic surface flaws," IEEE Trans. Neural Netw., vol. 6, no. 5, pp. 1201–1211, 1995

[4] Ge, Y., Fitzpatrick, J.M.: On the "Generation of Skeletons from Discrete Euclidean Distance Maps". 1996 1055–1066

[5] H. S. Chang and H. Yan, "Analysis of stroke structures of handwritten chinese characters," IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 29, no. 1, pp. 47–61, Feb. 1999

[6] J. J. Zou and H. Yan, "Skeletonization of ribbon-like shapes based on regularity and singularity analyses," IEEE Trans. Syst. Man. Cybern. B, Cybern., vol. 31, no. 3, pp. 401–407, 2001

[7] Lam, S.W.Lee, and C.Y.Suen, "Thinningmethodologies - Acomprehensive survey," IEEE Trans. Pattern Anal. Mach. Intell., vol. 14, no. 9, pp. 869–885, 1992
[8] Mayya, N., Rajan, V.T.: Voronoi Diagrams of polygons: "A framework for Shape representation". Proc. of 1994 638–643

[9] Sebastian, T.B., Klein, P.N., Kimia, B.B.: Recognition of shapes by editing their shock graphs. 2004**,** 550–571

[10] https://atoms.scilab.org/toolboxes/SIVP

[11] www.youtube.com/watch?v=aYDx25kTFrA

[12] http://sourceforge.net/p/sivp/discussion/456401
/thread/959f

[13] http://homepages.inf.ed.ac.uk/rbf/HIPR2/thin.htm

[14] http://homepages.inf.ed.ac.uk/rbf/HIPR2/thin.htm

[15]http://natureofcode.com/book/chapter-10-neural-networks/
[16] R. Gonzalez and R. Woods Digital Image Processing, Addison-Wesley Publishing Company, 1992, pp 518 - 548

AUTHOR:

First S. Thaminmun Ansari born in Arakkonam Town, Vellore District, Tamil Nadu, India in 1981, Completed his Bachelor of Engineering in electronics and & instrumentation at St Peters Engineering College under Madras university in 2004 and completing Master of Engineering in Embedded system Technologies at Anna university, CEG Campus, Guindy, Chennai, India by 2015.

He stared his career as “Embedded Programmer” at SANDS instruments Pvt Ltd. in 2004 and served till 2007. Then he switched to “Visteon Technical and Service Centre Pvt. Ltd”, Chennai, Tamil Nadu, India as Automotive software tester in 2007 and served till 2014 with various positions. Now he is working as Deputy Manager for Automotive software functional testing team in Pricol Limited, Coimbatore, Tamil Nadu, India. His research interest includes Image processing, Testing automation, OS programming etc