# Design and Implementation of an Autonomous 3D-printed Eye-Controlled System using Raspberry-Pi

*By*

***Hummam Ahmed Musa***

**Supervisor**
**Dr. Montassar Aidi Sharif**

*A thesis submitted in partial fulfilment of the requirements for the degree of*

*Computer Engineering Techniques*

*Computer Engineering Techniques Department*

Technical Engineering College

Northern Technical University

BSc

May 11th 2022

Kirkuk, Iraq

## Acknowledgements

I’d like to thank my advisor **Dr. Montassar Aidi Sharif** for his instrumental efforts in this work and my family for their supports and patience during my work in this project.

## Abstract

In order to help physically disabled persons to make their life independent or to make the humanoid robots real especially in Iraq, this paper proposes an autonomous eye-controlled system (AECS) on a Robotic head. In this work, several OpenCV image processing algorithms are employed to track the eye motion to coordinate the red colour moving left, right, and straight forward. We use the Raspberry-Pi 3 board as the system canter to process the images and control the motors. This project used the 3D printed technology to create the frame of the eyes and the framework of the project.

Experimental results show that the ACES system can be effectively used in the prototype and outperforms the identifications of the red colours and can track them more easily.

# Table of Contents

# List of Figures

# List of Abbreviations

| | |
|---|---|
| AECS | Autonomous eye-controlled system (AECS). |

# Chapter 1 Introduction

Today, autonomous  robotic eyes (3D - printed eyes) s are enhancing physical abilities of the elderly and disabled and making their life more accessible. This paper presents a novel implementation of an autonomous system for the completely disabled persons, which allows them to control  robotic eyes (3D - printed eyes) s by eye movement.

As the system structure shown in Figure 1, a camera is mounted in front to capture the image of any one of the eyes (either left or right) and track the position of eye pupils with the use of many image processing algorithms. According to the position of the eye, wheel chair motors are directed to move left, right, and forward.

The main part of the hardware system is the Raspberry-Pi board, which is utilized to perform the image processing and control the hardware system. It captures the frames of images in real time and extracts the commands from the eye motions. Then, the Raspberry- Pi sends the control signals to engines to perform the specific operation, such as running the motors in clockwise direction, anticlockwise, and stop the motors. On the  robotic eyes (3D - printed eyes) , an ultrasonic sensor for obstacle detection is additionally implanted for safety. It can notify the Raspberry-Pi and help to stop the motors in the event that an obstacle near the  robotic eyes (3D - printed eyes)  is detected.

The most challenge part of this system is the eye-motion detection algorithm. An open computer vision (OpenCV) library with numerous computer vision algorithms is used for the face and eye recognition [11]. For example, the component location calculation is used to identify single or numerous faces and both eyes. To find the correct eye pupil and its inside point is extreme objective of this algorithm. A few applications and calculations are further employed to discover the precise pupil area, and several sensor based recognition strategies, such as ECG, EEG, and EOG, are utilized to consequently discover eye understudy and follow eye pupils. The main contributions of this system are:

—We propose a software-hardware cooperative system for the eyemotion

controlled robotic eyes (3D - printed eyes) , in order to help physically handicapped people to make their life independent.

—Several image and video processing algorithms in view of face, eye, and eye pupil motion identification with least deferral of time are presented. Prototyping results show that the AECS system reduces the processing latency (less than 3 seconds) to 75% compared with the hand gesture controlled system proposed in [10] (4 seconds).

—Safety issue is further considered with the highest priority in this work to avoid the occasion of any collision. An ultrasonic sensor is implanted to detect the obstacles and stop the motion of the robotic eyes (3D - printed eyes) automatically.

## 1.1 Problem Statement

This project works with 3D design. The use of 3D printing technology was used to print the outside of the device and the rest of the pieces. This device relies on its movement on the x-axis and is responsible for the movement of the eye to the right and left And the Y axis is responsible for the movement of the eye up and down. Equipped with this eye as a mind or as a controller to control and process the images that the eye captures, the appropriate decision for the movement of the eye is based on what the camera lens captures, as it was programmed in the Python language and using the Open CV technology to track and isolate any red color that appears when following this name, and it can also be programmed to track any other color.

## 1.2 Thesis Organisation

The rest of the article is structured as follows: Chapter 2 presents related work and Chapter 3 discusses our proposed system architecture involving both hardware and software. In Chapter 4 we present the experimental work in this project. We demonstrate the system and evaluate the approach with simulation data in Chapter 5. Last, we summarize our work in Chapter 6.

# Chapter 2 Literature Review

The Currently, there are many control systems creating specific applications for individuals with different issues and incapacitates, such as the hand gesture controlled system [10], infrared light [15], ultrasonic and body kinecatics [3], and so forth [7, 4]. The drawbacks in these systems are: 1) they cannot be used by people of higher disability because they require accurate controls; 2) the infrared light based method [15] gives exact identification of the eye understudy focus area and additionally tracks the eye development. The infrared radiations, however, may influence the users' perceivability.

Moreover, systems based on voice recognition have been widely used today [13, 2, 5]. The main objective of these systems is that move the robotic eyes (3D - printed eyes) in particular direction based on the voice command. However, in some noisy conditions these systems may fail due to the difficulty for user to convey voice message to the controllers.

To overcome the aforesaid issues, the eye control system gives the freedom to make their life simple and helpful. It detects eye pupils' movement by using some motion sensors, such as ECG, EEG, and EOG [1], and a set of image processing arithmetics to control the robotic eyes (3D - printed eyes) . First of all, the wavelength of the white portion is recorded by the eyeball sensor. Then, when the user need to move right side, his left eye demonstrates no variety in wavelength however in the left eye the dark bit is detected by the sensor, which prompts diminish in the wavelength. Similar component happens in the right eye as well.

In what follows, many image processing algorithms from the OpenCV library [11] are utilized for face and eye motion discovery. For instance, the haar cascade arithmetic [12] is used to detect single or multiple faces and both eyes. To automatically find out and track the eye pupils' movement, the object detection [14], edge detection [9], and pattern recognition [8] are further employed in this study.

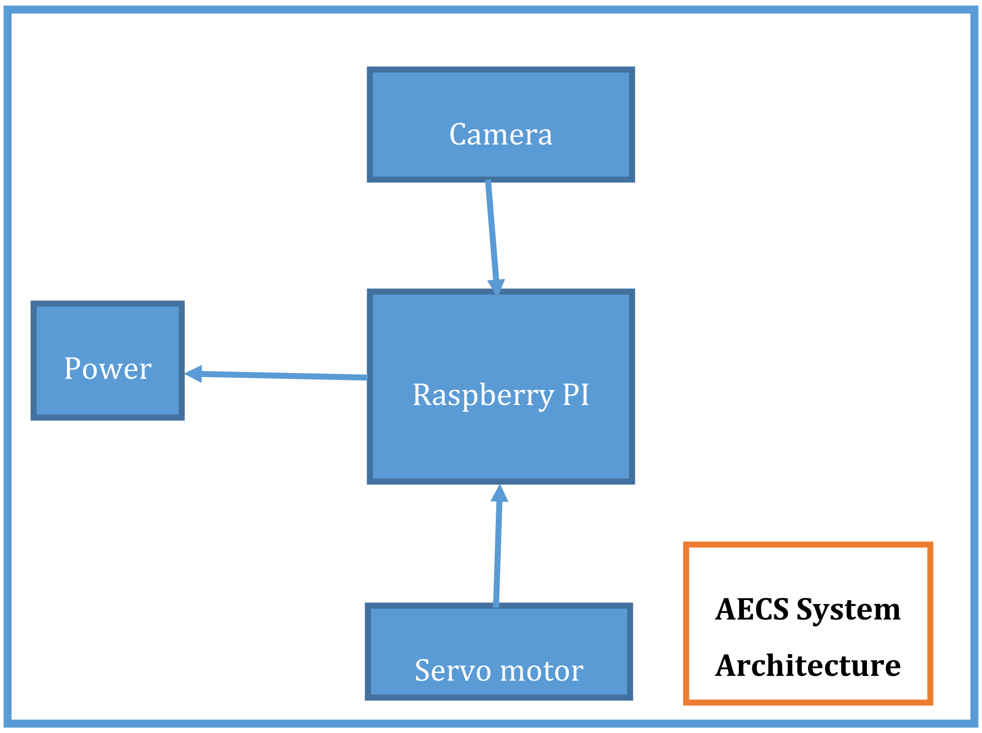


Figure 1:  The system structures

# Chapter 3 SYSTEM ARCHITECTURE

## 3.1 Hardware Structure

Figure 1 shows the hardware architecture with all the main components, including power supply, Raspberry-Pi board, cameras, motors, and so on. We also have a Wi-Fi module which helps in connecting the Raspberry-Pi to Internet and communicating using Internet in some emergency situations but in our work we did not use this facility because our work is limited. The power circuit gives the possible power supply to individual segments, involving the Raspberry-Pi, camera, sensor, and engines.

**Second,** the system is implemented based on real-time data acquisition. A low power consumption Raspberry-Pi board, as shown in Figure 2(a), is used as the control canter of the whole system. It provides input & output pins, USB, UART, HDMI ports and Ethernet adapter port for connecting it through Internet via wired or wireless connection using Wi-Fi adapter [18].

The Raspberry-Pi has a Broadcom BCM2835 framework on a chip, which incorporates an ARM1176JZF-S 700 MHz processor with a video core GPU and is initially transported with 512 MB of RAM and capable of up to 32 GB external memory. It works very efficiently with multiple images provided by the web camera connected with the Raspberry-Pi board. Distance between eye and the camera is 10 to 14 cm. The camera interfaces to the board using the UV4L driver.

In this work, not only the applications, but also the system performance is considered [16, 17]. Hence, a 720P video format camera is used as a balance among the memory cost, processing speed, and recognition accuracy. Figure 3(b) shows our prototyping demo of the auto-controlled robotic eyes (3D - printed eyes) . The motor driving circuit is connected with the Raspberry-Pi, the power supply of motors for controlling the motor driving integrated circuit. The Raspberry-Pi canter continuously generates command signals to enable the GPIO pins and performs the operations of left, right, forward, and stop.

Furthermore, safety issues is highlighted in this work. As shown in the Figure, there is an infrared sensor mounted in front of the robotic eyes (3D - printed eyes) to recognize the obstacles and stop the  robotic eyes (3D - printed eyes)  with the highest control priority.

## 3.2 Software Design

The Raspberry-Pi has its own operating system – Raspbian. It runs python programs on Linux where it can be associated with primary board. The Win32 disk imager software is used in this work to boot a Raspberry image file. While putting a bootable memory device on Raspberry-Pi board, it can access the Raspbian directly.

The main part of the image processing algorithms is completed with the assistance of the OpenCV 3.0.0 library [11]. OpenCV is released under a BSD license and it is thus free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS, and Android. It is designed for computational efficiency and with a strong focus on the real-time applications. Enabled with OpenCV, our proposed work can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

## 3.3 System functioning

The AECS system begins with catching pictures consistently from a web camera. By analysing the captured images, the system identifies the red colour all conceivable circle displayed on the specific area. In what follows, the separation between the canter other colours and the red colour area is measured using the coordinate system logic.

Figure 4 shows the operational flow chart. It is organized into blocks having an internal structure formed from the three fundamental elements: a state box, a decision box, and a conditional box. State boxes are rectangles, conditional boxes are rectangles with round corners, and decision boxes are diamond-shaped.

The state box represents the status of the system, involving Initial state, Obstacle Detection state, and Move Detection state in this flow chart. The values in the decision boxes determine the possible paths through the block under the action of the inputs. For example, if there is no developments of the eye, then it goes back to the Initial state. In the state of Obstacle Detection, the wheelchair must be immediately stopped in the event that an obstacle is detected near to the wheelchair, since the safety issues has the highest priority in this system design. If there is no obstacle detected, the flow chart goes into the Move Detection state. In this state, both engines move when the eye is in canter, and the wheelchair moves in forward bearing. Likewise, when eye development is left, a wheelchair left side engine runs; when eye recognition is right, the right motor runs.

For the begin and stop operations, eye flickering to control the wheelchair is used. In the case that the eye shut for 3 seconds, the system totally stops and at the end of the day it will close the eye for 3 seconds, the system is then reactivated.

# Chapter 4 Experimental Work

## 4.1 Device Requirements

Having built a simple single-eye mechanism in the past, I wanted to improve on the design as well as making it more accessible to the maker community. The updated assembly uses parts that can easily be bought online, and almost all the components can be printed easily without supports. Designing the model in this way does sacrifice some functionality, but I'll be releasing an optimised design in the future. This project is ideal if you want to build a functional and realistic eye mechanism, but don't necessarily have access to tools like a lathe or speciality components.

Another feature of this design is that it's designed to use snap-in eyes which can be replaced and used with other 3-D printed components to cast a highly realistic dome over the painted eye. This process is quite involved so I have another instructible on how to make the eyes, but if someone prefer to use simple 3-D printed eyes can do that too.

This project will require some post processing of your prints, including some (hand) drilling and sanding, but other than the basics (3D printer, craft knife, screwdriver, Allen keys) there are no special tools needed.

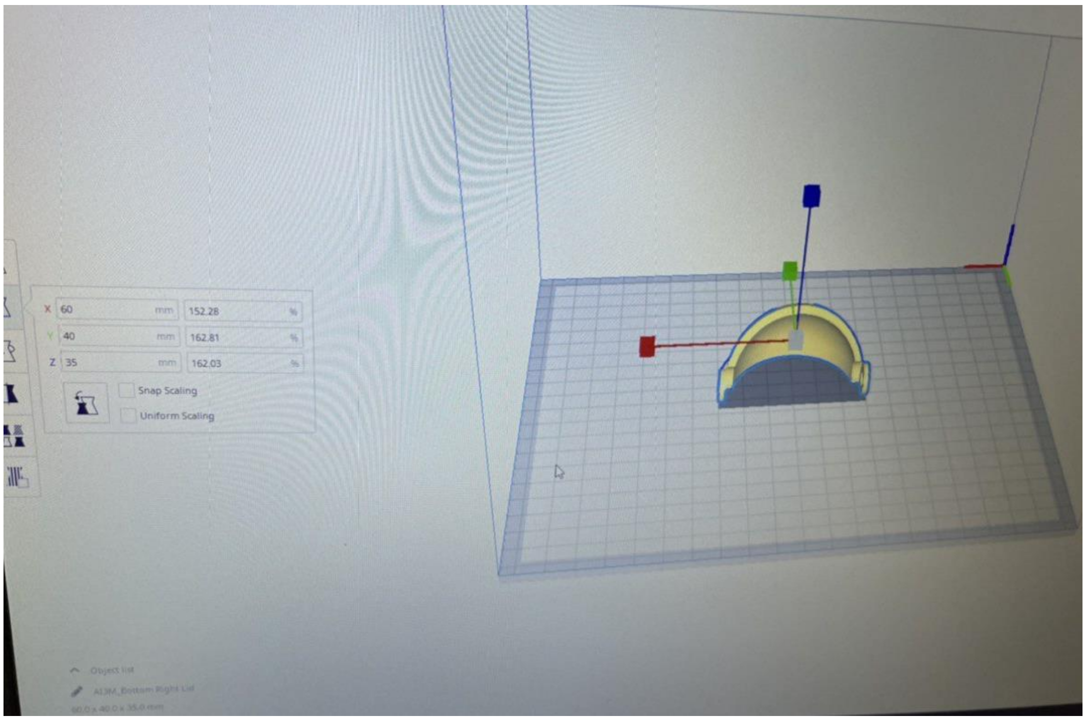
## 4.2 Materials and Components:

1. 3D Printer Filament: PLA is fine although I'd recommend you use a good brand because some parts are quite small and fragile. ABS is good for making realistic eyes but not necessary.
2. 6x SG90 Micro Servos
3. Various M2 and M3 screws although any screws roughly that size should work ok.
4. Arduino: This design was tested using a genuine Uno, but it's likely that any board which has SDA/SCL pins, 3 analogue inputs and a digital input will work.
5. Servo Driver Board: I opted for a 16 channel PWM driver board
6. 5V Power Supply, around 4A is more than enough.
7. A female DC power jack to match your power supply, to be soldered to the servo driver board
8. Jumper Cables
9. Joystick
10. Potentiometer (10k ohms is generally a good value to use)
11. Momentary switch (Some joysticks have this built in, but its easier to control when its separate.
12. 10k Resistor.

## 4.3 Work Procedure

### Printing

Printing may be somewhat challenging due to the small parts, but the majority of the parts print quickly and easily without supports. We used PLA for all my parts other than the eyes (which were ABS as it looks a bit more natural). There are a few delicate parts to look out for as well, but if someone using decent quality filament and you're happy with your print settings you should be fine. Finally, We used a layer height of 0.2mm and this was more than precise enough for this model - We suspect anybody could get away with 0.3mm even.



```
X  60   mm   152.28  %
Y  40   mm   162.81  %
Z  35   mm   162.03  %
   [ ] Snap Scaling
   [ ] Uniform Scaling
```

Figure 2: Printing area

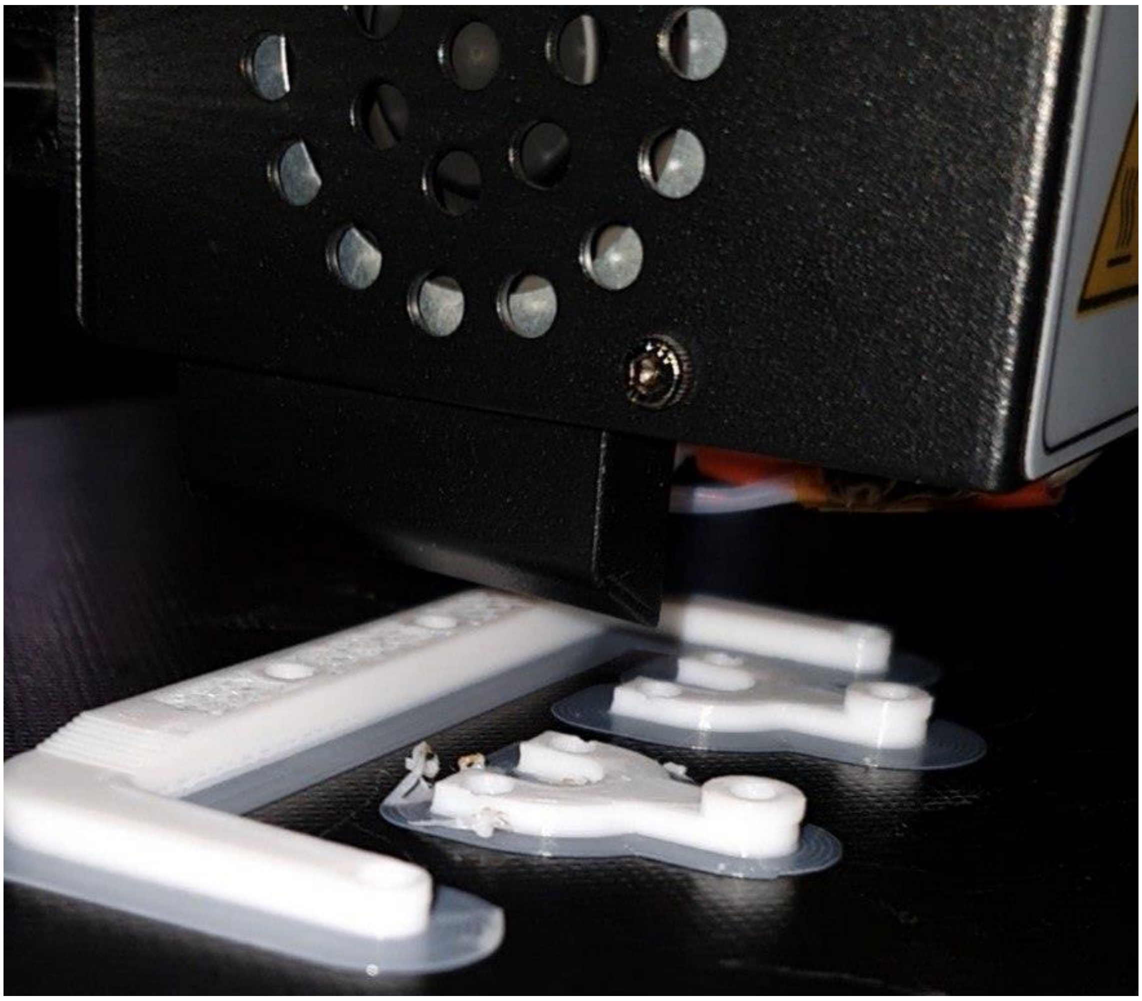


Figure 3: Parts in the 3D printer

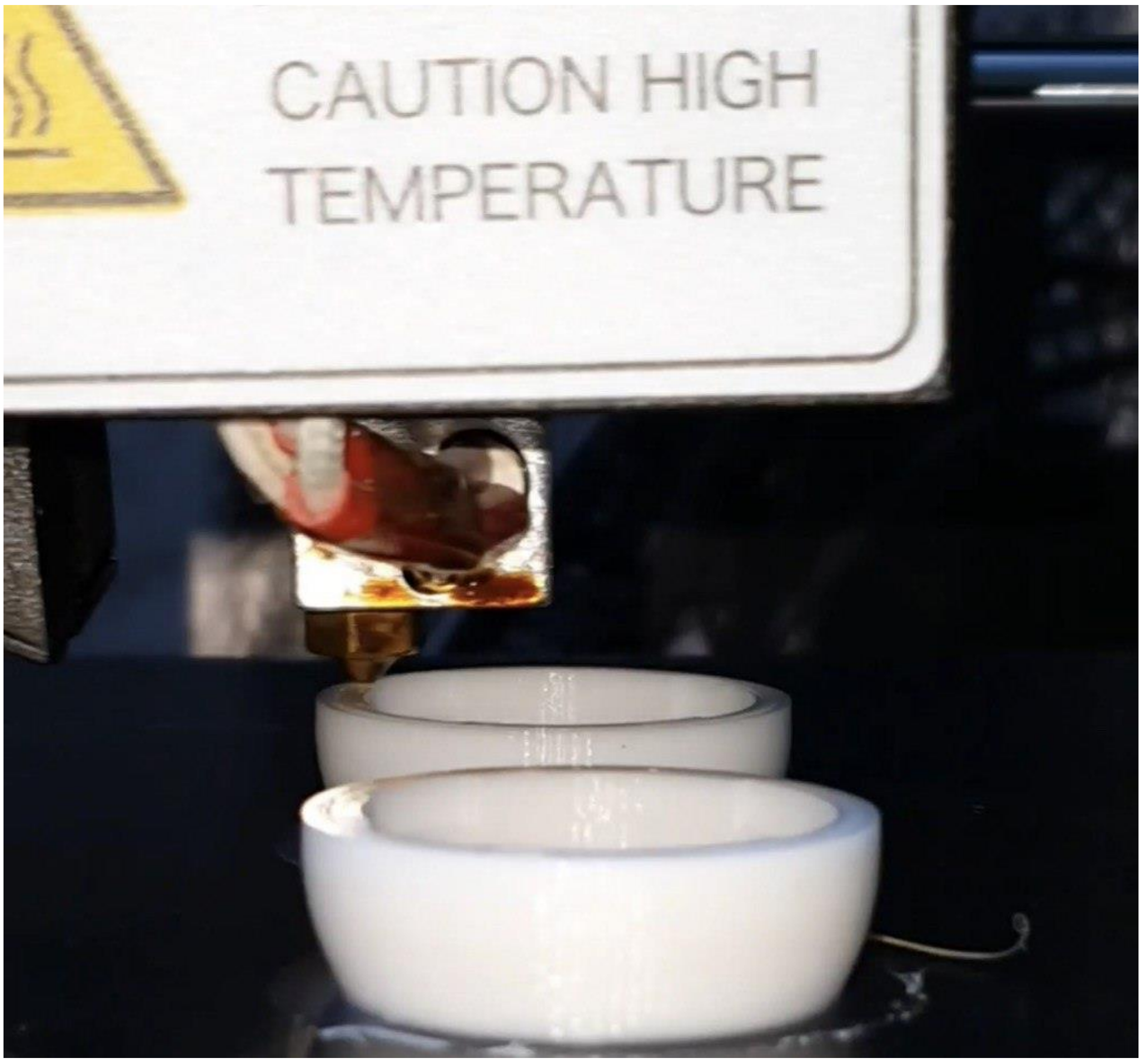


Figure 4: 3D-Printing parts

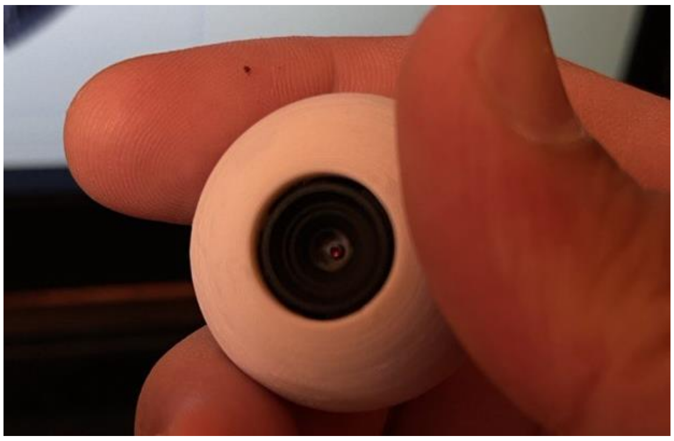


Figure 5:  3D-Printing eyes

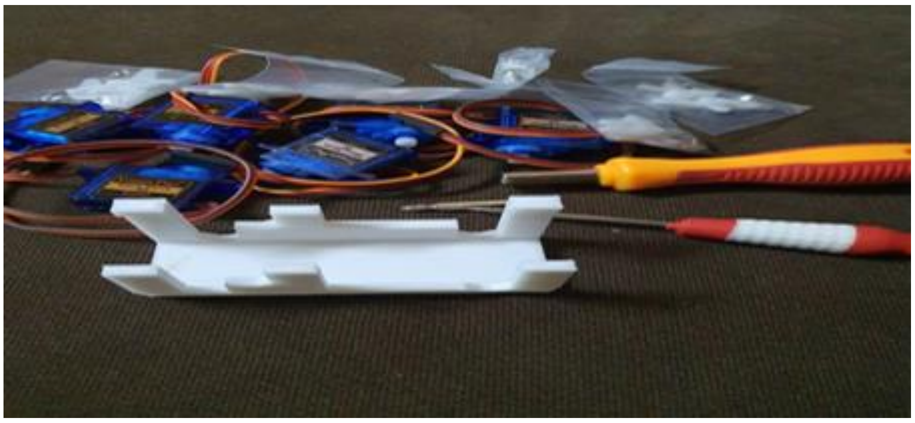


Figure 6: Several parts

## Post-Processing

The eyes can be sanded and smoothed as much as you like, but a minimum of sanding may be necessary on the bottom of the model where the printer would have been printing an overhang. This is just to ensure the eye can rotate smoothly within the eyelids. The only other parts that we sanded were the eye adaptor components, just because they are a tight fit and the bottom few layers on my prints always tend to be a bit swollen.

The parts are designed to print such that some holes are undersized enough to be directly screwed into, whereas others are oversized enough so the screw will pass through them snugly. If your printer is making the holes to small to screw into or rotate smoothly around however you can use a little hand drill to drill out some of the holes to make them more precise, and tapping threads is an option too (although PLA usually grips screws fairly well anyway). Check the images for a guide as to which holes should be which size.

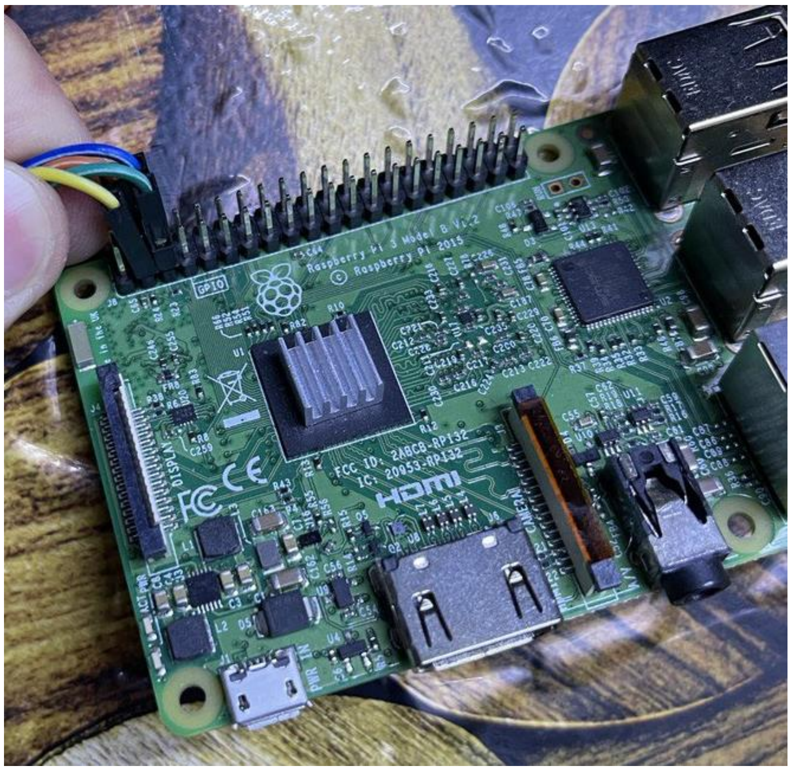


Figure 7: Raspberry Pi

## Assembly

Once all your parts are printed and processed, you can assemble your model! It may be helpful to refer to the video to see how it all goes together. Also there's all the reference pics in one folder in my download, including an stl of the complete model you can look at.

1. Connect the two bases with 10mm/12mm M3 bolts, this pivot point is for the y-axis of the eye motion and the eyelids.
2. Place the servo in position and screw it in with some 4 or 6mm M2 screws, this serves as the actuator for the x-axis motion
3. Attach the y-axis arm to the sub-base with a 4/5/6mm M3 screw, and attach a servo horn on the third hole from the centre using a 4mm or 6mm M2 screw. Check above to make sure the orientation of everything is right.
4. Start building the x-axis assembly by screwing the forks into the eye-adaptors with 4/5/6mm M3 bolts, the fork holes should be oversized so the

screws bite into the adaptor, one goes in on a funny angle but you should be able to get it in.

5. Attach the three-point connector to the top of the forks, the M3 screw will bite into the undersized hole in the fork component. Also attach a servo arm on the final hole to the centre of the three-point connector using a 5mm M3 bolt (the hole on the servo arm will likely need to be drilled to 2.5mm - 2.8mm to accept the screw). I'd recommend manipulating the assembly to make sure it all moves okay without friction regularly as you build it up too.
6. Attach the eye centre-link to the eye adaptors with a 8mm M3 screw, make sure the flat surface of the centre-link is facing up and the sloping section facing down. You can also plug in the eyes at this stage.
7. Screw all this to the centre of the sub-base with two 8/12mm M3 bolts.
8. Load up the servo block with 5 TowerPro SG90 servos, in the correct orientation shown.
9. Work out which eyelid is which using the graphic, and connect the relevant connector with a 4mm or 6mm M2 screw, and attach a servo arm to the other end (use the last hole in the servo horn - you may need to drill this to 1.5mm - 1.8mm).
10. Attach the eyelids to the base, but don't worry about connecting any servo horns yet.

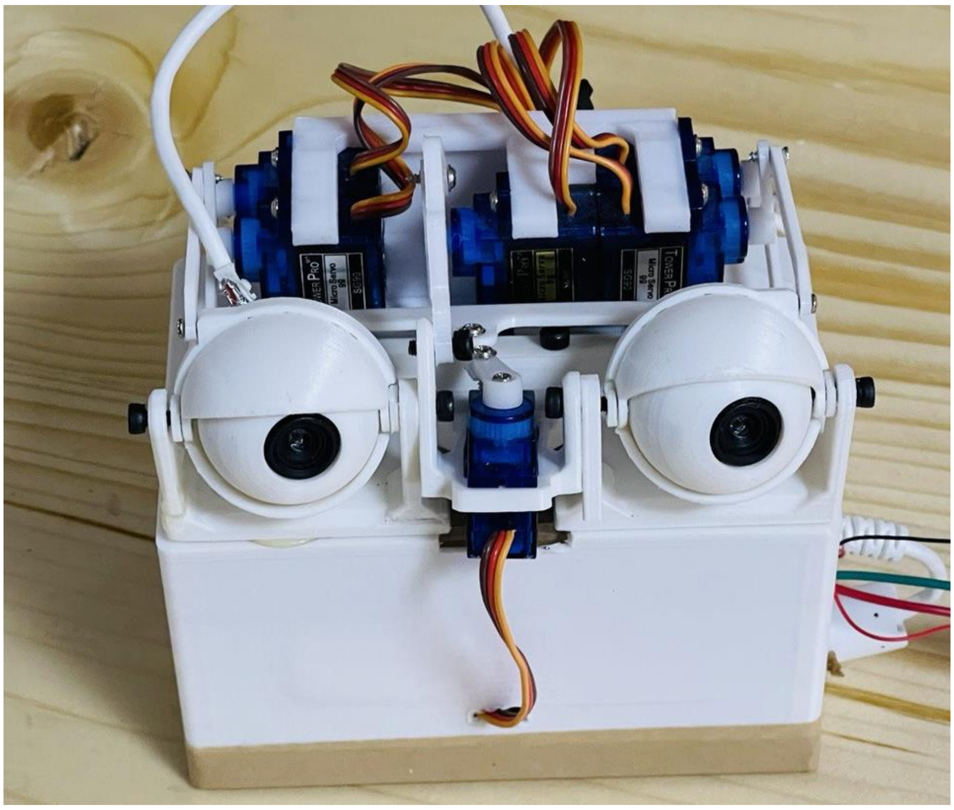


Figure 8: The complete system

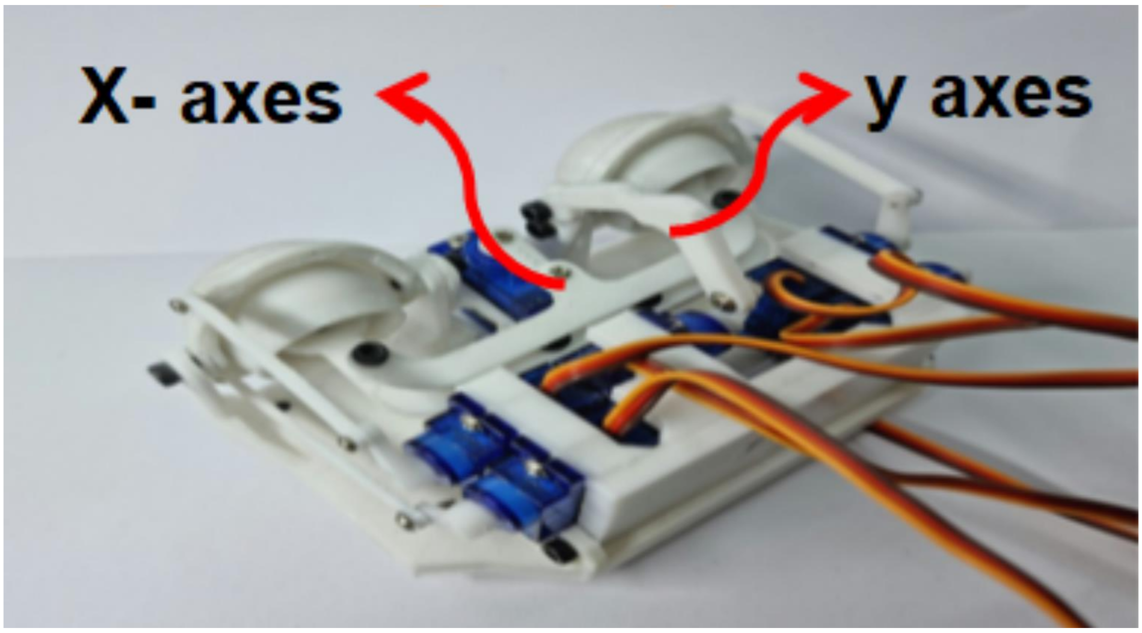


Figure 9: The complete system axes

# Chapter 5 Results and Discussion

## 5.1 Results

By referring to the graphic below, upload the code to the Raspberry and wire everything up. Check the up the servo driver board. All servos should now be powered and in their neutral position, so use this opportunity to link up all the servo arms to the servos with the eyes facing straight forward in a neutral position. One can just plug them in, then disconnect the power to screw them in properly. The y-axis servo arm is in an awkward position to accept a screw.

In this project, We'd recommend testing the motion with your joystick at this stage to make sure there are no issues.

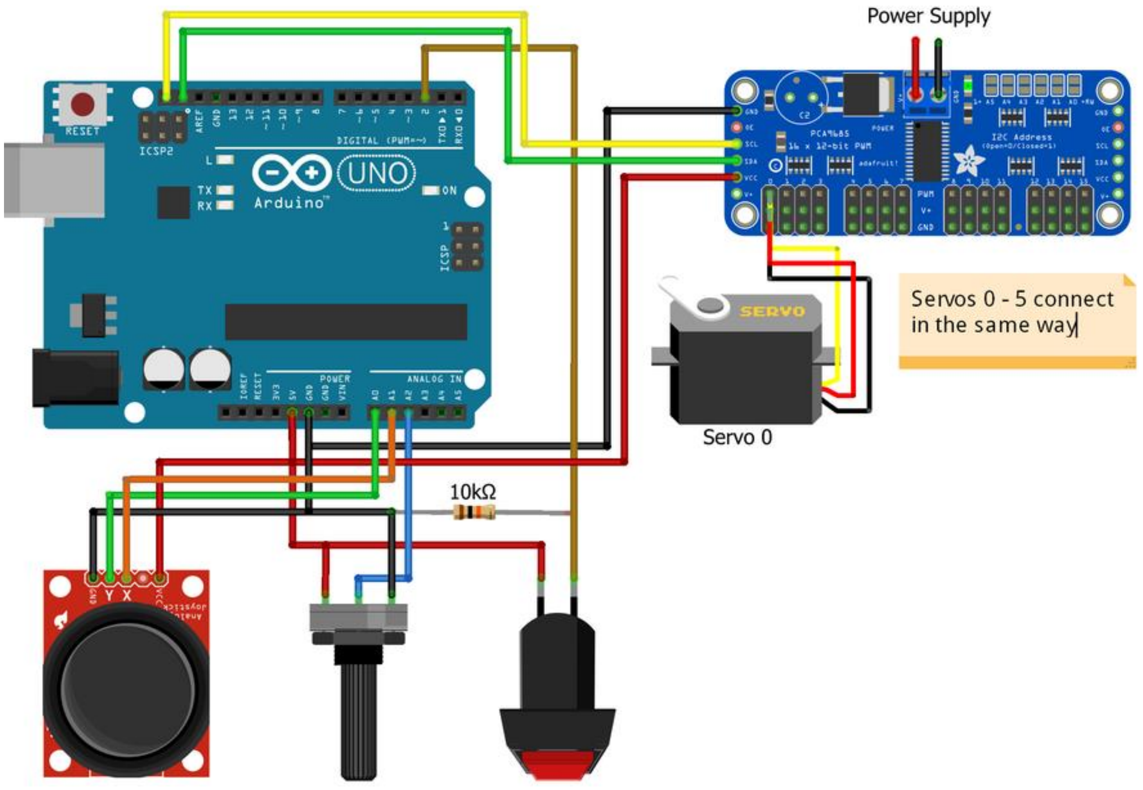


Figure 10: Wiring

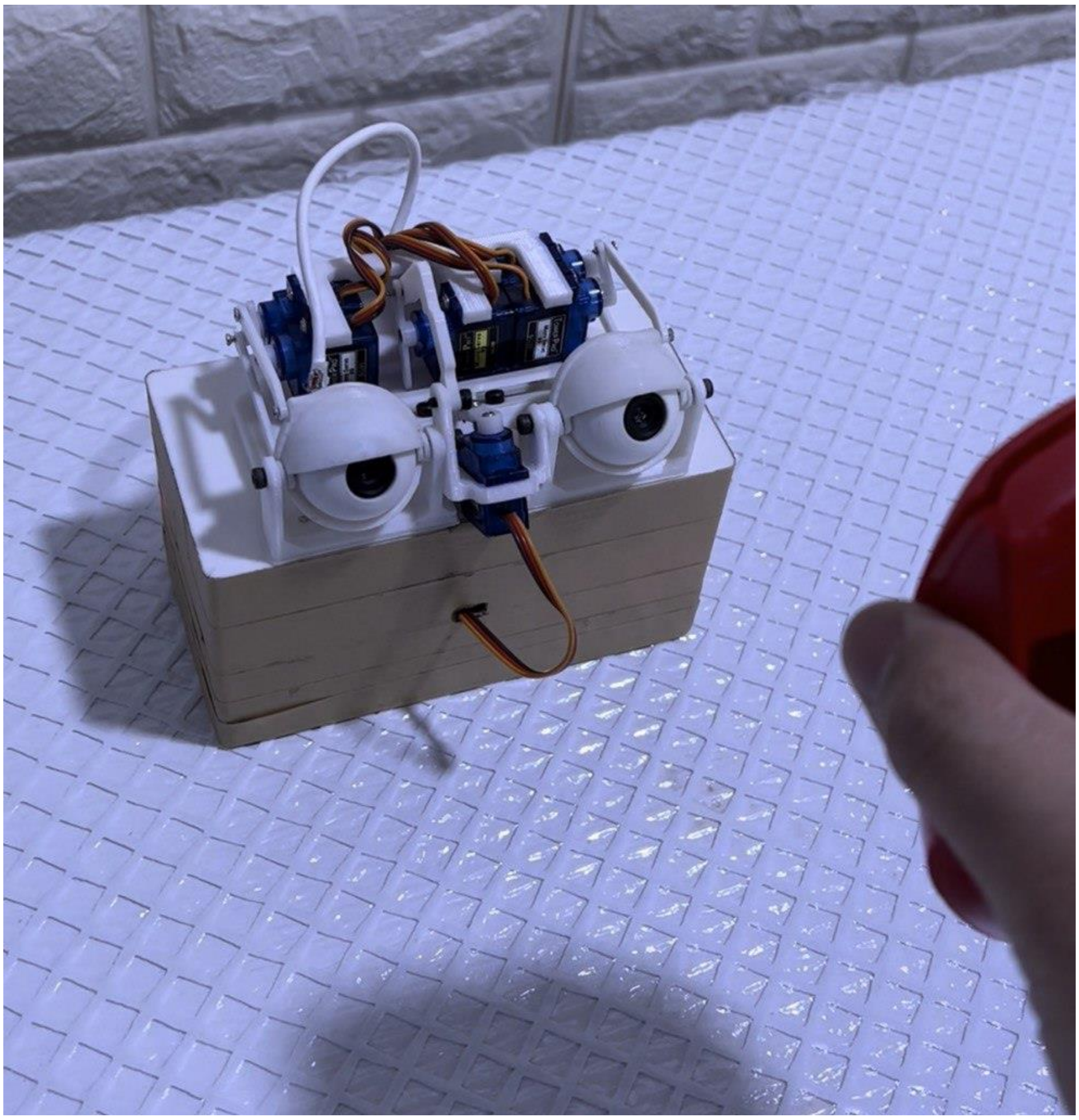


Figure 11: Testing the device

For the eyelids, its best if you set the servos to be in the blinking position so you can line them all up in the centre. We would do this by either holding down the blink switch or creating a short over it. Once all the servo arms are in position, it’s easy to screw them in.

## 5.2 Discussion

This project proposes a software-hardware co-design system on 3D-printed automated eyes to reduce the need of help required in the robotic applications and with the handicapped individuals. Several digital image processing algorithms from the OpenCV library are employed in this work, such as background subtraction, RGB to grayscale conversion. Experimental results show that our proposed AECS system can be effectively used on the the 3D-printed eyes prototype..

However, the system performs the 3D-printed robotic eyes movement operation with 3-second delay time. It is not acceptable as an commercial usage. In addition, to track the red colour in the low-light condition is another big challenge for the AECS system. That is one reason that the other auto-controlled 3D-printed robotic eyes usually applied very complex and combined systems with multiple technologies to improve the detecting accuracy, although sometimes it is a conflict with the system speed and resource cost.

# Chapter 6 Conclusions and Recommendations

## 6.1 Conclusion

This study suggested an autonomous eye-controlled system (AECS) on a robotic head to assist physically challenged people in becoming self-sufficient or making humanoid robots a reality, particularly in Iraq. Several OpenCV image processing methods are used to track the eye motion to coordinate the red colour going left, right, and straight forward in this work. To process the images and control the motors, we use the Raspberry-Pi 3 board as the system controller. The frame of the eyes and the framework of the project were created using 3D printing technology. Experiments reveal that the ACES method may be employed well in the prototype, outperforming red colour detection, and allowing for easy tracking.

## 6.2 Recommendation

In the future, therefore, instead of processing algorithms with Raspbian, FPGA-based implementations such as RGB to Grayscale and Edge Detection will be employed on gate-level chips in order to improve the speed..

## References

[1] A. Ahamed, A.U. Ahad, H.A. Sohag, and M. Ahmad. Development of low cost wireless biosignal acquisition system for ecg, emg and eog. 2015 2nd International Conference on Electrical Information and Communication Technologies (EICT2015), December 2015.

[2] C. Aruna, A. Dhivya Parameswari, M. Malini, and G. Gopu. Voice recognition and touch screen control based wheel chair for paraplegic persons. 2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE2014), March 2014.

[3] P. Britto, J. Indumathi, S. Sivarasu, and L. Mathew. Automation of robotic eyes (3D - printed eyes) using ultrasonic and body kinematics. National Conference on Computational Instrumentation (NCCI2010), pages 19–20, March 2010.

[4] A. Dev, H.C. Chacko, and R. Varghese. Eye controlled wheel chair using eog. International Conference on Computing and Control Engineering (ICCCE 2012),, April 2012.

[5] M.A. Devi, R. Sharmila, and V. Saranya. Hybrid brain computer interface in robotic eyes (3D - printed eyes) using voice recognition sensors. 2014 International Conference on Computer Communication and Informatics (ICCCI2014), Janauary 2014.

[6] E.S. Gedraite and M. Hadad. Investigation on the effect of a gaussian blur in image filtering and segmentation. 53rd International Symposium ELMAR-2011, pages 393 – 396, September 2011.

[7] S.U. Kumar and V.M. Vinod. Eog based robotic eyes (3D - printed eyes) control for quadriplegics. 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS2015), March 2015.

[8] Y. Li, X. Xu, N. Mu, and L. Chen. Eye-gaze tracking system by haar cascade classifier. 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA2016), June 2016.

[9] P. Melin, C.I. Gonzalez, J.R. Castro, O. Mendoza, and O. Castillo. Edge-detection method for image processing based on generalized type-2 fuzzy logic. IEEE Transactions on Fuzzy Systems, 22(6):1515–1525, January 2014.

[10] S.D. Mhaske and M.P. Dale. Hand gesture controlled robotic eyes (3D - printed eyes) system using image processing. International Journal of Electrical, Electronics and Computer Systems, 4(2):2347–2820, February 2016.

[11] OpenCV. Open source computer vision. http://opencv.org/.

[12] M. Rezaei, H.Z. Nafchi, and S.Morales. Global haar-like features: A new extension of classic haar eatures for efficient face detection in noisy images. 6th Pacific-Rim Symposium on Image and Video Technology (PSIVT 2013), 8333:302–313, 2013.

[13] M.F. Ruzaij, S. Neubert, N. Stoll, and K. Thurow. Design and testing of low cost three-modes of operation voice controller for robotic eyes (3D - printed eyes) s and rehabilitation robotics. 2015 IEEE 9th International Symposium on Intelligent Signal Processing Proceedings (WISP2015), May 2015.

[14] K.S. Sabarish and A.M. Suman. Eyeball and blink controlled robot with fuzzy logic based obstacle avoidance system for disabled. International Conference on Electrical and Electronics Engineering (ICEEE2012), pages 16 – 17, June 2012.

[15] Y.G. Shin, K.A. Choi, S.T. Kim, and S.J. Ko. A novel single ir light based gaze estimation method using virtual glints. IEEE Transactions on Consumer Electronics, 61(2):254–260, July 2015.

[16] X. Yang and J. Andrian. A low-cost and high-performance embedded system architecture and an evaluation methodology. 2014 IEEE Computer Society Annual Symposium on VLSI (ISVLSI2014), pages 240–243, March 2014.

[17] X. Yang and J. Andrian. A novel bus transfer mode: Block transfer and a performance evaluation methodology. Integration, the VLSI Journal, 52(C):23–33, January 2016.

[18] X. Yang, X. Niu, J. Fan, and C. Choi. Mixed-signal systemon- chip (soc) verification based on system verilog model. The 45th Southeastern Symposium on System Theory (SSST 2013), pages 17–21, March 2013 2013.

[19] Bing Z. Using vector quantization of hough transform for circle detection. 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA2015), pages 393 – 396, December 2015.