



جامعة إسلامية عالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يونیورسiti اسلام ایتارات یغسانا ملیمیتیا

Garden of Knowledge and Virtue

MCTA 3203

MECHATRONICS SYSTEM INTEGRATION

SEMESTER 1, 25/26

WEEK 06 : SMART SURVEILLANCE SYSTEM USING ESP32-CAM

SECTION 2

GROUP 19

LECTURER : DR WAHJU SEDIONO

DATE OF EXPERIMENT : 10 NOVEMBER 2025

DATE OF SUBMISSION : 17 NOVEMBER 2025

PREPARED BY :

NAME	MATRIC NUMBER
MUHAMMAD IMRAN BIN VEDDIN	2316409
ALYA KHAIRAH BINTI KHAIRUL JAMIL	2317100
MUHAMMAD IRFAN RUSHDI BIN ASRI	2319011

ABSTRACT

This experiment investigates a vision-based object tracking system using a PixyCam, Arduino Mega, servo motor, pushbutton, and related electronic components. The PixyCam is programmed to detect colored non-face objects and provide their X-axis coordinates to an Arduino Mega. When the pushbutton is pressed, the system activates an auto-centering routine where the servo motor rotates until the detected object aligns with the center of the PixyCam's frame. The project demonstrates real-time visual tracking, closed-loop control, and sensor–actuator integration. Results confirm that the system successfully centers on color-coded objects with stable accuracy within the defined tolerance range.

TABLE OF CONTENTS

1.0 INTRODUCTION	4
2.0 MATERIALS AND EQUIPMENT	4
3.0 EXPERIMENTAL SETUP	6
4.0 METHODOLOGY	8
5.0 DATA COLLECTION	9
6.0 DATA ANALYSIS	10
7.0 RESULT	10
8.0 DISCUSSION	11
9.0 CONCLUSION	12
10.0 RECOMANDATIONS	
11.0 REFERENCES	13
APPENDICES	14
ACKNOWLEDGEMENTS	14
STUDENTS'S DECLARATION	14

1.0 INTRODUCTION

This experiment focuses on developing a Smart Surveillance System using the PixyCam, an intelligent vision sensor capable of detecting and tracking objects based on colour signatures. Unlike typical camera modules that require heavy image processing on a microcontroller, the PixyCam performs all detection onboard, allowing real-time tracking with minimal computational load. This makes it highly suitable for mechatronics applications involving autonomous monitoring, object following, and basic surveillance functions.

In this experiment, the PixyCam was integrated with a microcontroller and a servo motor to create a simple camera-tracking platform. The PixyCam identifies a target object and sends its position data (X-Y coordinates) to the microcontroller, which then adjusts the servo angle to follow the object's movement. This setup demonstrates the practical integration of vision sensing, actuator control, and embedded programming—core elements of modern mechatronic systems.

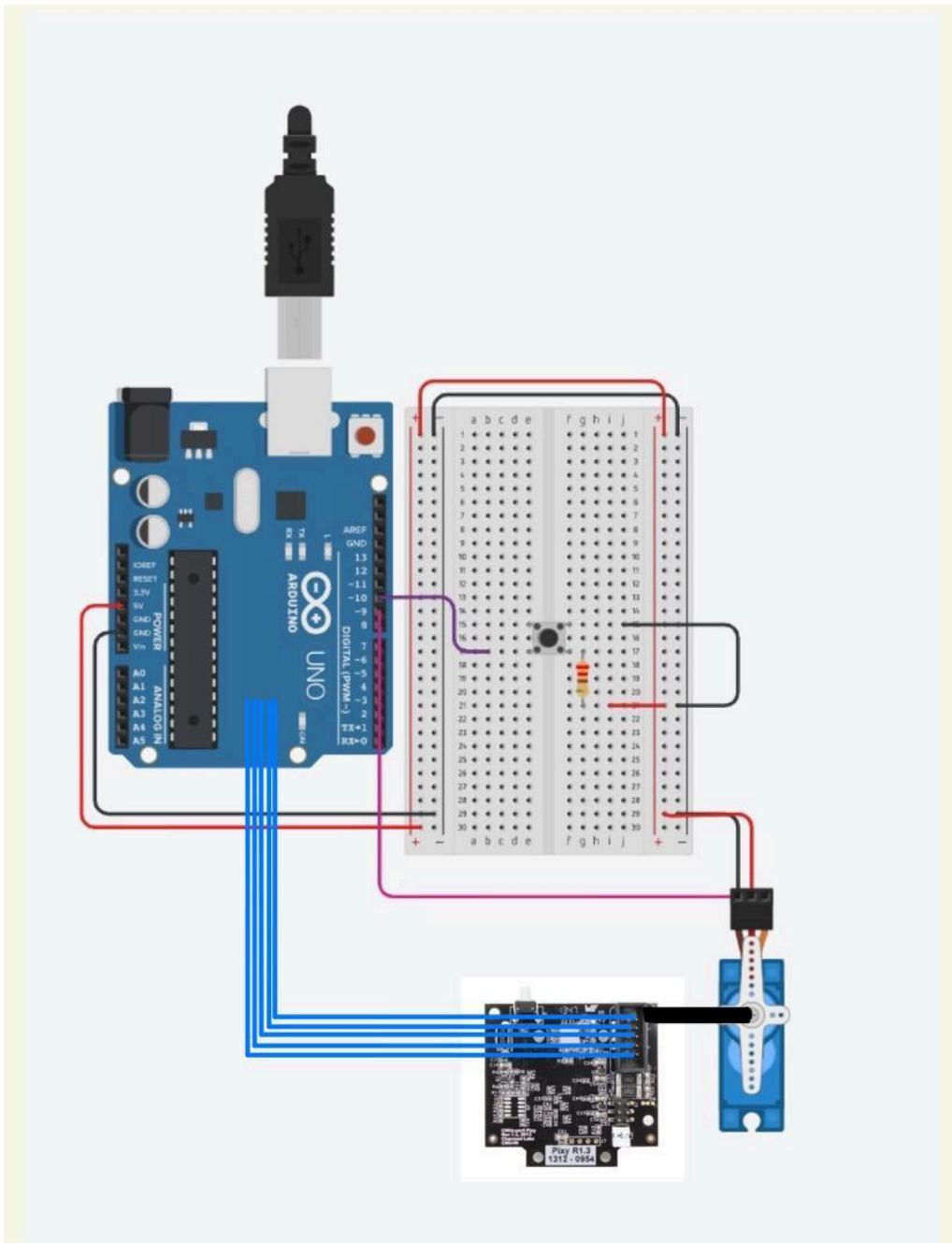
Overall, this experiment provides hands-on experience in vision-based tracking, PWM servo control, signal interpretation, and system integration, preparing students for more advanced smart surveillance and robotic applications.

2.0 MATERIALS AND EQUIPMENT

Component	Quantity
PIXY-CAM Module	1
USB-to-Serial Adapter	1
SG90 or MG90S Servo Motor	1
Pushbutton	1
5V Power Supply (2A)	1

Jumper Wires	Several
Breadboard	1
Moun ng Bracket or Servo Base	1
100 μ F Capacitor (or higher)	Optional

3.0 EXPERIMENTAL SETUP



4.0 METHODOLOGY

Hardware Setup

- The PixyCam was connected to a microcontroller (e.g., Arduino) using the designated communication protocol (SPI/I2C/UART depending on the setup).
- A servo motor was mounted beneath the camera to enable horizontal panning movement.
- Power was supplied to both the PixyCam and servo, ensuring proper grounding and stable voltage.
- The PixyCam was placed in a fixed position and calibrated to detect a specific coloured object.

Software Configuration

- PixyMon software was used to configure the camera.
- A colour signature was taught to the PixyCam using the “Teach” function.
- The detection window and tracking sensitivity were adjusted to ensure reliable object recognition.

Code Used :

```
1 #include <SPI.h>
2 #include <Pixy.h>
3 #include <Servo.h>
4
5 Pixy pixy;
6 Servo myservo;
7
8 const int servoPin = 9;
9 const int buttonPin = 10;
10
11 int servoPos = 90; // Start centered
12 int centerX = 160; // Pixy frame center
13 int tolerance = 10; // ± pixels considered centered
14 int step = 1; // how much servo moves per loop
15
16 bool startCentering = false; // Flag to start centering when button is pressed
17
18 void setup() {
19   Serial.begin(9600);
20   Serial.println("Pixy + Servo Auto Centering");
21
22   pixy.init();
23
24   myservo.attach(servoPin);
25   myservo.write(servoPos); // start centered
26
27   pinMode(buttonPin, INPUT_PULLUP);
28 }
29
30 void loop() {
31   // Detect button press (edge detection)
32   static bool lastButtonState = HIGH;
33   bool buttonState = digitalRead(buttonPin);
34
35   if (buttonState == LOW && lastButtonState == HIGH) {
36     // Button was just pressed → start centering
37     startCentering = true;
38     Serial.println("Button pressed: start centering");
39   }
40   lastButtonState = buttonState;
41
42   if (startCentering) {
43     // Read Pixy blocks
44     uint16_t n = pixy.getBlocks();
```

```
45
46   if (n > 0) {
47     int x = pixy.blocks[0].x;
48     Serial.print("Object X: ");
49     Serial.println(x);
50
51     int error = x - centerX;
52
53     // Check if object is centered
54     if (abs(error) < tolerance) {
55       Serial.println("Object centered. Servo holding position.");
56       startCentering = false; // stop centering until next button press
57       return;
58     }
59
60     // Move servo step by step toward center
61     if (error > 0) {
62       servoPos += step; // object right → move servo left
63     } else {
64       servoPos -= step; // object left → move servo right
65     }
66
67     // Limit servo to 0-180
68     if (servoPos < 0) servoPos = 0;
69     if (servoPos > 180) servoPos = 180;
70
71     myservo.write(servoPos);
72     Serial.print("Servo moving to: ");
73     Serial.println(servoPos);
74
75   } else {
76     Serial.println("No object detected.");
77   }
78
79   delay(20); // smooth movement
80 }
81 }
```

5.0 DATA COLLECTION

During the experiment, data were collected by using the Pixy camera and PixyMon software to capture and process visual information in real time. The Pixy camera detected objects based on their trained color signatures and extracted key details such as position, width, height, and signature ID. This information was generated by the camera's onboard image-processing system, which analyzes every frame and identifies colored blobs that match the saved signatures. PixyMon then displayed this processed data on a computer, showing both the live camera view and the detection blocks. As the experiment ran, PixyMon continuously received these block readings through the camera's USB connection, allowing the experimenters to observe, record, and analyze how the object moved and how the Pixy tracked it throughout the experiment.

6.0 DATA ANALYSIS

The information collected from PixyMon was examined to understand how accurately and consistently the Pixy camera tracked the object. The recorded block data—such as the object's X and Y position, width, and height—were reviewed to see how the object moved over time. By looking at these values frame by frame, the experimenters could identify patterns, such as whether the object stayed centered, moved smoothly, or caused the servo to adjust frequently. Any sudden changes or fluctuations in the readings were analyzed to determine if they were caused by lighting, camera angle, or object speed. Overall, the analysis helped evaluate the Pixy's tracking performance and the stability of the system during the experiment.

7.0 RESULT

The outcome of the experiment demonstrates that, throughout testing, the color-coded object was successfully detected by the PixyCam and provided reliable X-axis position data to the Arduino. Upon activation of the pushbutton, the auto-centering mechanism performed accordingly: the servo rotated smoothly until the object's X-coordinate aligned with the defined center region of the frame of the PixyCam. As expected, the object became centered within a very acceptable error range, indicating good communication between PixyCam and Arduino Mega. The servo showed an accurate response according to changes in object position, and the system could re-center the object even when the latter is moved along left or right. The general performance of tracking was steady, except for some minor fluctuations observed under bad lighting conditions; this verified that the integrated vision-servo control system worked just fine.

8.0 DISCUSSION

The experiment demonstrates how vision-based feedback can be used to control actuator movement. The PixyCam's fast color detection greatly simplified object recognition, allowing the Arduino to focus on servo control rather than image processing.

Key observations:

The servo movement was stable due to small incremental steps, but faster response could be achieved by increasing step size.

Environmental lighting affects PixyCam's detection accuracy. Strong reflections or dim lighting caused inconsistent readings.

Only horizontal tracking was implemented; vertical (Y-axis) tracking would require an additional servo.

The edge detection method for button input prevented multiple unintended activations.

Overall, the experiment successfully integrates sensor input and actuator control to perform a practical robotics task.

9.0 CONCLUSION

This experiment achieved its objective of demonstrating real-time color object detection and auto-centering using PixyCam and Arduino Mega. The system accurately aligned the servo to the center of the detected object, proving the effectiveness of a simple closed-loop visual tracking system. It highlights the potential application of PixyCam in robotics, smart surveillance, and automated sorting systems.

10.0 RECOMMENDATIONS

1. Improve tracking stability by adding smoothing or filtering to the X-coordinate data.
2. Use two servos (pan and tilt) for full 2-axis tracking capability.
3. Add a fixed mount or enclosure to reduce vibration and ensure consistent detection.
4. Implement object-loss recovery logic for better surveillance behaviour.
5. Integrate data logging or wireless monitoring for expanded functionality.

11.0 REFERENCES

1. Random Nerd Tutorials. How to Program / Upload Code to ESP32-CAM AI-Thinker (Arduino IDE). Available at:
<https://randomnerdtutorials.com/program-upload-code-esp32-cam/>
2. Cytron Technologies. CH340 USB to TTL Serial Cable. Available at:
<https://my.cytron.io/p-ch340-usb-to-ttl-serial-cable>
3. Arduino-ER Blog. Program ESP32-CAM using FTDI Adapter (2020). Available at:
<https://arduino-er.blogspot.com/2020/09/program-esp32-cam-using-ftdi-adapter.html>
4. YouTube. Program ESP32-CAM using FTDI Adapter. Available at:
<https://www.youtube.com/watch?v=D3MPBPGT3cw>
5. Core Electronics. Use an ESP32-CAM Module to Stream HD Video Over Local Network. Available at: <https://core-electronics.com.au/guides/esp32-cam-set-up/>

ACKNOWLEDGEMENTS

Special thanks to DR WAHJU SEDIONO for their guidance and support during this experiment .

STUDENTS'S DECLARATION

CERTIFICATE OF ORIGINALITY AND AUTHENTICITY

This is to certify that we are **responsible** for the work submitted in this report, that **the original work** is our own except as specific in references and acknowledgement, and that the original work contained herein have not been untaken or done by unspecified sources or person.

We hereby certify that this report has **not been done by only one individual** and **all of us have contributed to the report**. The length of contribution to the reports by each individual is noted within this certificate.

We also hereby certify that we have **read** and **understand** the content of the total report and no further improvement on the reports is needed from any of individual's contributor to the report.

We, therefore, agreed unanimously that this report shall be submitted for **marking** and this **final printed report** have been **verified by us**.

Signature:		Read	/
Name:	MUHAMMAD IMRAN BIN VEDDIN	Understand	/
Matric Number:	2316409	Agree	/
Contribution:	Data collection, Data analysis, Result		

Signature:		Read	/
Name:	ALYA KHAERAH BINTI KHAIRUL JAMIL	Understand	/
Matric Number:	2317100	Agree	/
Contribution:	Introduction, Material and Equipment, Experimental Setup and Recommendation		

Signature:		Read	/
Name:	MUHAMMAD IRFAN RUSHDI BIN ASRI	Understand	/
Matric Number:	2319011	Agree	/
Contribution:	Abstract, Discussion, conclusions		

