Robotic Arm for Monitoring and Ballistic Operations

Chad Lucas
School of Electrical Engineering and Computer Science
University of North Dakota
Grand Forks, USA
chad.lucas.ndus@edu.com

***Abstract*— This project builds and design a Ai based object detection and tracking defense system. It uses a Raspberry Pi for interfacing and control, Ai camera for detecting red objects, two LX-16A servo motors for pan and tilt operations allowing for object tracking, logic level shifters for accurate and safe signal communications between components, a lidar sensor for distance measurement which activates a DC motor if object is withing the minimum distance requirements. The system will accurately detect and track a red object and then fire a projectile once the target is within threshold distance. This system shows that with minimal cost and components you can provide a solid foundation to autonomous defense systems.**

Keywords—AI object detection, Raspberry Pi, servo motor control, Lidar, logic level N-Channel MOSFET.

I. Introduction

The fog of war can turn decision making into paralysis. In this chaotic environment it makes it difficult to identify targets and where every second counts making a wrong decision can increase not only costs but casualties as well. With the help of integrated AI based systems human error can decrease drastically as well as increase situational awareness. AI is transforming the battlespace, it allows for fast decision making and by continuous improvements in object detection models it allows for less human involvement, allowing commanders to focus on other missions.

This project will build and design an integrated AI object detection and tracking systems. It will use a Raspberry Pi as the controller interface along with the Raspberry Pi AI-camera. The camera will use built-in functions and object detection models to identify a red object and track the object in a 1280x720 frame in real-time. There will also be two LX-16A servo motors that provide the turret system allowing for tracking the object. The servos will get their input from the built-in camera functions which will be mapped to X-Y coordinates for the servo's movement. The system also will use a TF-Luna Lidar sensor to measure the object's distance and set a threat distance of 20 cm or less. If the target enters the 20 cm radius the sensor will send a signal to the Pi's GPIO pin to set it high, triggering the MOSFET controlled DC motor (firing mechanism). This system demonstrates that a minimum hardware and software setup can provide robust security measurements.

II. Methodolgy

A. *Ai camera detection*

The object detection utilizes built in Raspberry PI AI camera libraries for color masking and coordinate extraction. These functions allow us to create bounding boxes around the target of interest and return its location based on the overall framing size. For our system further processing was needed to improve the accuracy of our tracking arm. By finding the center of the bounding box our arm is able to track the object as shown in Figure 1.

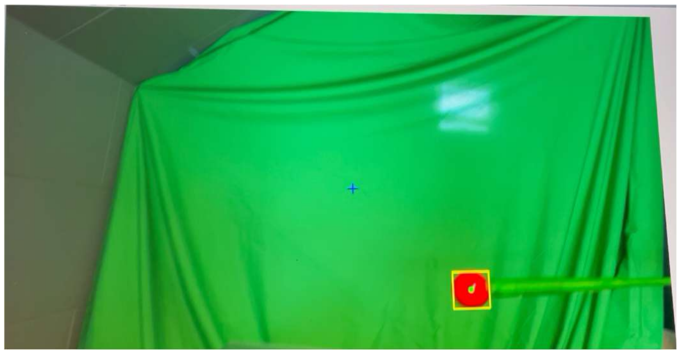


Figure 1: Yellow bounding box created by Ai camera module

B. *Servo motor cotrol*

- Mapping

For the object tracking and servo movement range we first had to establish our coordinate system. We used a frame width of 1280x720, this gave us an optimal tracking area that wouldn't overload the Pi with unnecessary frame information which would result in reduced processing power and would reduce the response time of our servo motors.

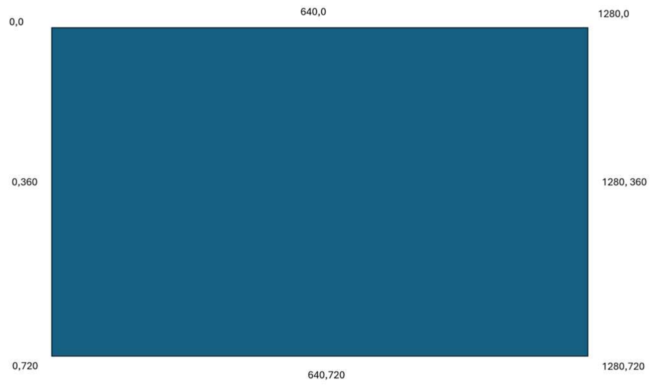


Figure 2. 1280x720 Frame for coordinating mapping.

Figure 2. Coordinate system of the AI camera showing image resolution (1280×720). The top-left corner represents (0, 0) or NW, and the bottom-right corner is (1280, 720) or SE. The object center (640, 360) corresponds to the middle of the frame

Since our servo have a range from 0-1000, we have to translate our 2D frame area to our servos and also take into account our system will be inverted based on the physical setup of the arm. To get the center of our target we use:

- servo_pan=$(1-\frac{X}{Width})\times 1000$
- servo_tilt=$(1-\frac{Y}{Width})\times 1000$

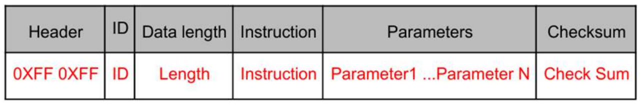
These equations result in a smooth proportional translation from frame to servo. The multiplication by 1000 is our scaling factor that normalizes the result to keep our values withing the servo ranges.

- LX-16A Servo Motors

The use of the LX-16A servo motors were chosen for their built in communication protocols. The LX-16A is controlled by serial port commands [1]. This gives us more precision with less complicated mapping than normal Pulse width modulated servos. Two servos were used for the pan and tilt operations and given unique ID assignments to communicate over the shared serial bus.

- ID 6: pan servo, responsible for horizontal movement
- ID 4: tilt servo, responsible for vertical movement.

The UART serial protocol used to communicate and control the servo motors consists of command packet with a set of bytes as shown [1]:



Header	ID	Data length	Instruction	Parameters	Checksum
0XFF 0XFF	ID	Length	Instruction	Parameter1 ...Parameter N	Check Sum

Figure 3: LX-16A Servo Packet Format. The packet includes a header, ID, length, command, data fields, and checksum for validation.

- Header: Start of the operation
- ID: Servo ID
- Data length: Number of bytes from the command
- Instructions: The operation function of the servo
- Parameters: Supplemental instructions.
- Checksum: verifies total number of bytes sent.

To communicate between the Pi and servo motors we set the baud rate at 115200 and created python functions to send correctly formatted packet data. Some of the functions include reading the current position of the servo, moving the servo to the position of the object, getting the voltage and getting the temperature of each servo. For example, to move the servo:

```
def move_servo(id, position, time=0):
    pos_L = position & 0xFF
    pos_H = (position >> 8) & 0xFF
    packet = [0x55, 0x55, id, 7, 1, pos_L, pos_H,
              time & 0xFF, (time >> 8) & 0xFF]
    checksum = (~sum(packet[2:]) & 0xFF)
    packet.append(checksum)
    ser.write(bytearray(packet))
```

Figure 4. Python function to send position command

C. *Logic Level Shifting*

The Raspberry Pi operates at 3.3V logic level and our servos motors operate on a 5V logic level. In order to provide safe and adequate voltage levels for our systems to operate reliably we need to add a logic level shifter. We used a TXS0108E logic level shifter to convert the servos 5V to the 3.3V that the Pi's GPIO pins operate at.

D. *Lidar*

The Tf-Luna lidar sensor was used to measure the distance to our object. The Time of Flight (ToF) principle was used to measure the distance with periodically emitting infrared modulated waves. The sensor calculates the time by measuring the phase difference between the original wave and the reflection wave as shown in Figure 5 [2].

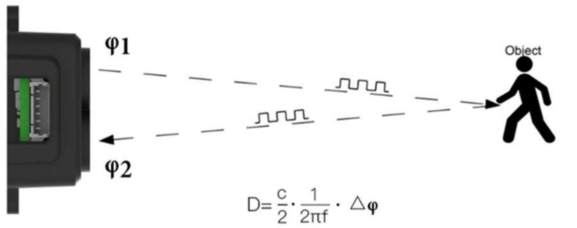


Figure 5 showing distance (D) calculated by measuring the phase difference.

Since our servos motors are operating on the UART serial communications bus the lidar sensor will use the I2C protocols, in order to ensure our sensor is operating we will make sure pin 5 is set to ground based on the datasheet configuration as shown in Figure 6.

No.	Function	Description
1	+5V	Power supply
2	RXD/SDA	Receiving/Data
3	TXD/SCL	Transmitting/Clock
4	GND	Ground
5	Configuration Input	Ground: I2C mode Disconnected/3.3V: Serial port Communications mode
6	Multiplexing output	On/off mode: Output I2C mode: Data ready signal

Figure 6 shows the pin connection and pin 5 for I2C mode.

The sensor will continuously monitor the distance and compare it to our proximity threshold in our case 20 cm radius. If the sensor detects motion within our designated limit, it will send a signal to set the Pi's GPIO pin to high and trigger our firing mechanism as shown in Figure 7 .

E. *Actuator Control*

The Raspberry Pi GPIO pins only output a small current which isn't sufficient to drive the dc motor to activate the firing mechanism. In order to provide enough current to the motor a logic-level N-Channel MOSFET (IRLZR44N) was used. The gate of the MOSFET is connected to limiting resistor which in turn is connected to the GPIO pin, the drain is connected to the negative terminal of the DC Motor and the source is connected to ground. This MOSFET transistor acts as a switch driving the dc motor when the gate voltage is turned on. This allows us to automate the firing mechanism with the addition of our lidar sensor which acts as our control.

We can see from Figure 7 , our LTSPICE schematic that our system operate with minimum components and hardware.

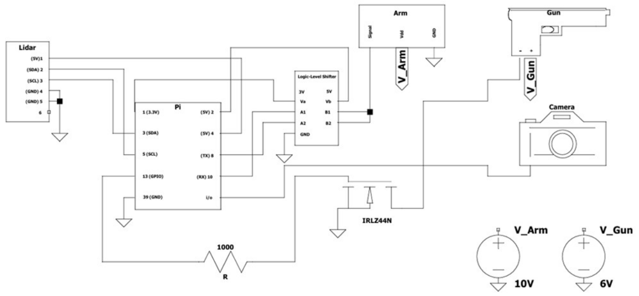
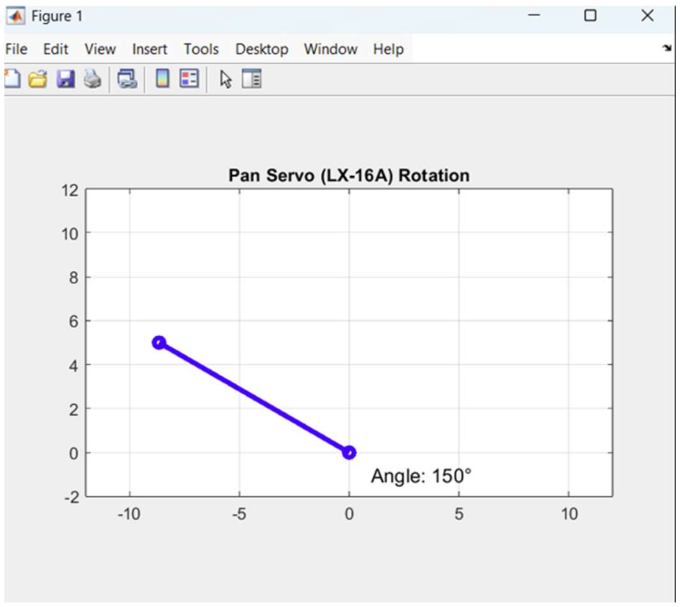


Figure 7. Ltspice schematic showing the complete object tracking circuit.

IV.Simulation

To verify the angle and ranges of the servo motors of the LX-16A, we used MATLAB to model the servo from 0°-180°. A 2D coordinate system was used with a variable angle shown in Figure 7-1.



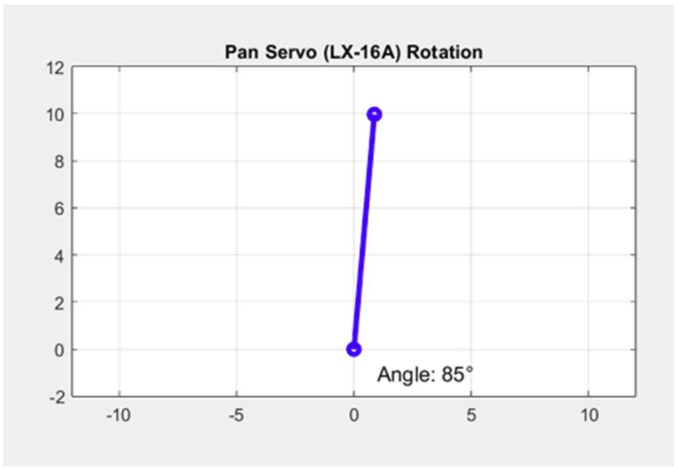


Figure 7-1 Shows 2D rotation of simulated servo arm.

To determine the arms direction we will use basic trigonometry equations where L is the length of our arm:

- $X = L * \cos(\theta)$
- $Y = L * \sin(\theta)$
- $radians = \theta * \frac{\pi}{180}$

To simulate the behavoir of our LiDAR sensor which calculates the distance form the sensor to the target object we used the distance formula and plotted the results as shown in figure 7-2 :

$$d = \sqrt{(x_{target} - x_{sensor})^2 + (y_{target} - y_{sensor})^2}$$

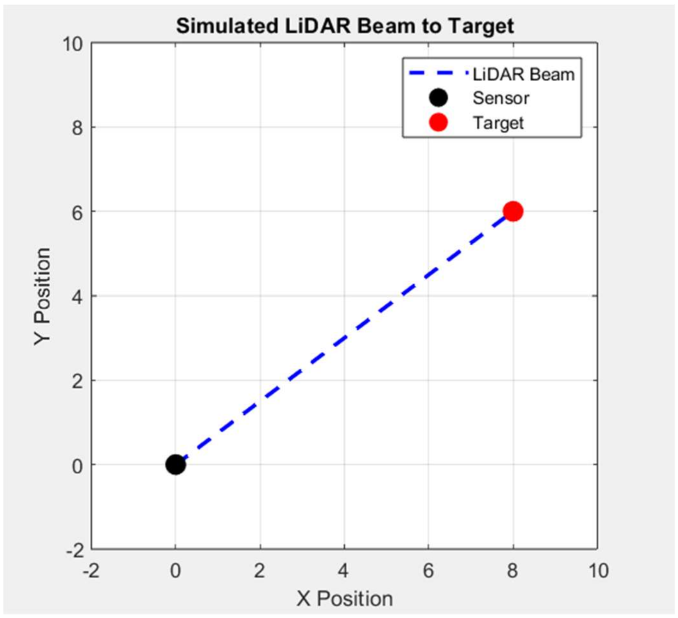


Figure 7.2. LiDAR simulation plot in MATLAB.

V.Results

The AI object detection and targeting system was tested in a controlled indoor environment. Each subsystem was verified for functional performance. A red object was used as the target for detection and tracking. The following results were observed

- The AI camera successfully identified the red object and created a bounding box in real time.
- The LX-16A pan and tilt servos properly responded to the inputs and provided the correct angle and aimed at the object consistently.
- The TF-Luna accurately provided real-time distance tracking and ensured to engage the actuator if the object was within the minimum distance threshold of 20 cm.
- The N-Channel MOSFET (IRLZ44N) provided enough current to drive the DC motor when triggered via GPIO.

As we see from Figure 8 the firing pin is activated when the red object is within 20 cm radius.

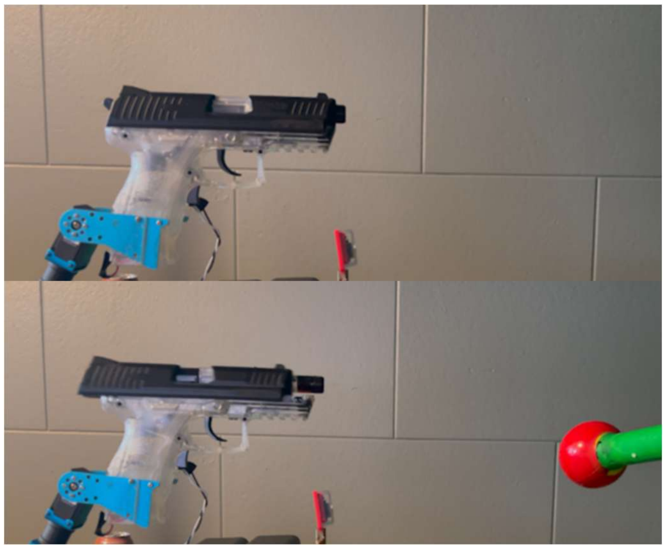


Figure 8. Firing rod activated when object w/in 20cm radius.

IV.Challenges

During initial development and testing, several constrains, and challenges were encountered.

- **Signal Voltage Mismatch**: Since the servo motors operate on 5V logic and the Raspberry Pi operates on 3.3V the need for a logic level shifter was determined to be needed for safe and reliable operations.

- **GPIO Current limitation**: Initial conditions when using a npn BJT transistor were used provided unreliable operations. Changing the transistor to a MOSFET logic level transistor provide enough current to drive the DC motor reliably.

- **Servo Mapping**: Initial movement was inaccurate and unstable. By accurately mapping and constraining the servo motors to the 2D camera frame and adjusting the internal speed of the LX-16A servos, we were able to achieve an accurate tracking system.

V. Improvements

Several improvements and additions can be implemented to enhance our system:

- **Object Classification**: Adding more realistic object classifications that can identify in real time threating movements vs non- threating movements.

- **Wireless Communication Protocols**: Implementing safety standards and adding wireless communications to our system to easily identify friendly or enemy targets.

- **Mobile Unit**: Integrate the detection and tracking system onto a mobile unit to move to the detected target, making it more useful in a realistic environment.

VI. Conclusion

This project aimed to implement and design a real-time AI detection and tracking system using a Raspberry Pi, AI Camera, Lidar sensor, servo motors and a DC motor (actuator). The system was composed of a few subsystems such as the ai object detection system, the servo tracking system, the lidar distance system and the firing system.
The raspberry Pi Ai camera module detects the red object and calculates the coordinates based on its position of the frame, which then gets mapped to the coordinates for the LX-16A pan and tilt servos. The TF-Luna Lidar sensor then continuously measures the objects distance and makes sure it is 20cm or greater. If the object enters the sensor threshold it activates the DC motor to fire the projectile.
To make the system more robust and operate in a realistic environment several issues need to be addressed and improved upon. For instance, adding a filter to the lidar sensor to reduce noise levels. Also implementing better object detection algorithms instead of a just a single tone color mask.

Our system showed that a real time object and target firing system can be implemented with very few components and power consumption. With this system it can be used to aid in the defense sector. In wartime environments it is often noisy and chaotic making it difficult to identify enemy or friendly targets and where every second counts. This Ai detection and firing system aims to reduce human error and uncertainty on the battlefield.

REFERENCES

[1] [1] LewanSoul, “LX-16A Bus Servo User Manual,” [Online]. Available: https://www.lewansoul.com/support

[2] Benewake, “TF-Luna LiDAR Sensor Datasheet,” [Online]. Available: https://wiki.dfrobot.com/TFmini_LiDAR_Module_SKU_SEN0259

[3] Raspberry Pi Foundation, “Raspberry Pi 5 GPIO Pinout,” [Online]. Available: https://www.raspberrypi.com/documentation/computers/raspberry-pi.html

[4] TXS0108E Logic Level Shifter Datasheet, Texas Instruments, [Online]. Available: https://www.ti.com/product/TXS0108E

[5] IRLZ44N N-Channel MOSFET Datasheet, International Rectifier, [Online]. Available: https://www.infineon.com/cms/en/product/power/mosfet/n-channel/irlz44n/

[6] Python Software Foundation, “pyserial Documentation,” [Online]. Available: https://pyserial.readthedocs.io/

[7] OpenCV.org, “Color Detection and Masking with OpenCV,” [Online]. Available: https://docs.opencv.org/elps make faster and more accurate decisions on the battlefield.

[8] https://www.armyupress.army.mil/Journals/Military-Review/Online-Exclusive/2024-OLE/Multidomain-Battlefield-AI/