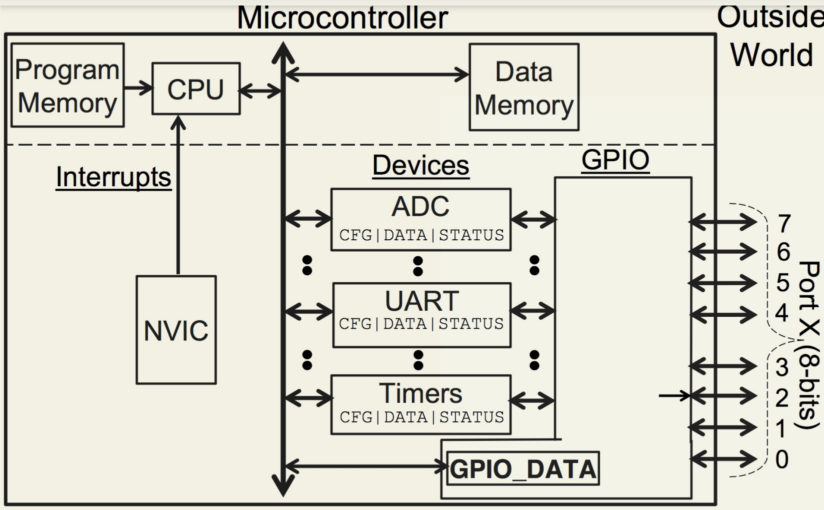
CprE 288 – Introduction to Embedded Systems (Object Detection and Analysis)

Instructor:
Dr. Diane Rover

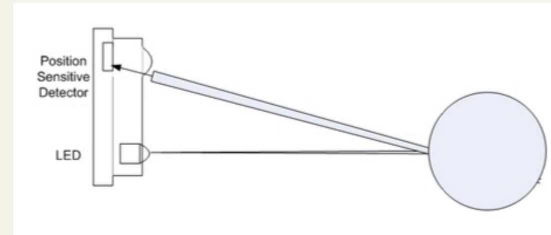
Concepts

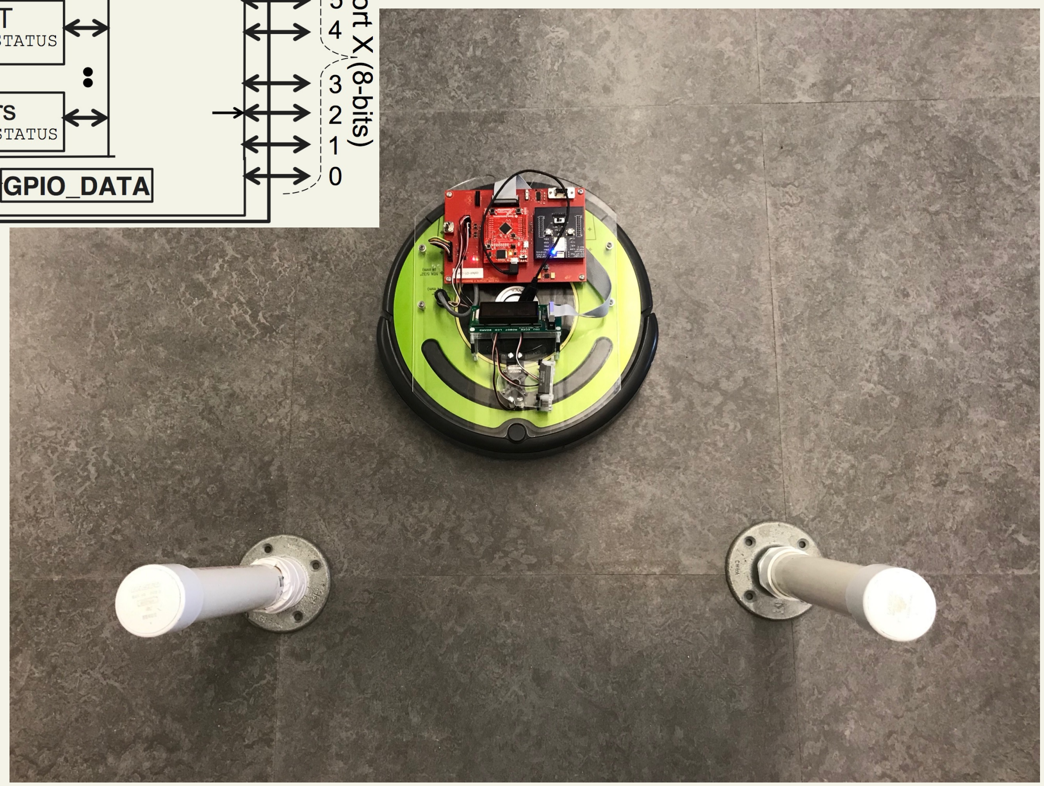
- hardware/software system
- system integration
- code review
- object detection
- object analysis (e.g., features)
- valid sensor data
- object width
- arc length

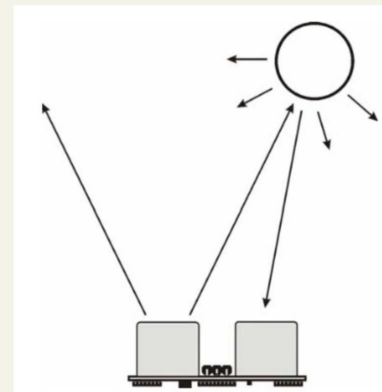
Object Detection and Analysis



Lab 7

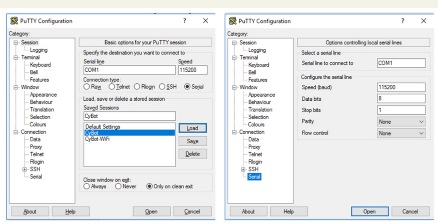


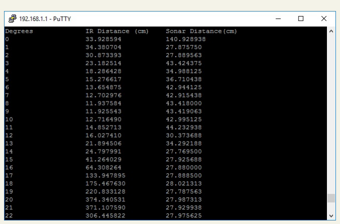


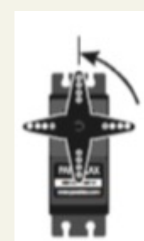


Lab 8

Labs 5 and 6







Lab 9

Older Lab: https://www.youtube.com/watch?v=qyUFNEJBems

Block Diagram for TM4C MCU

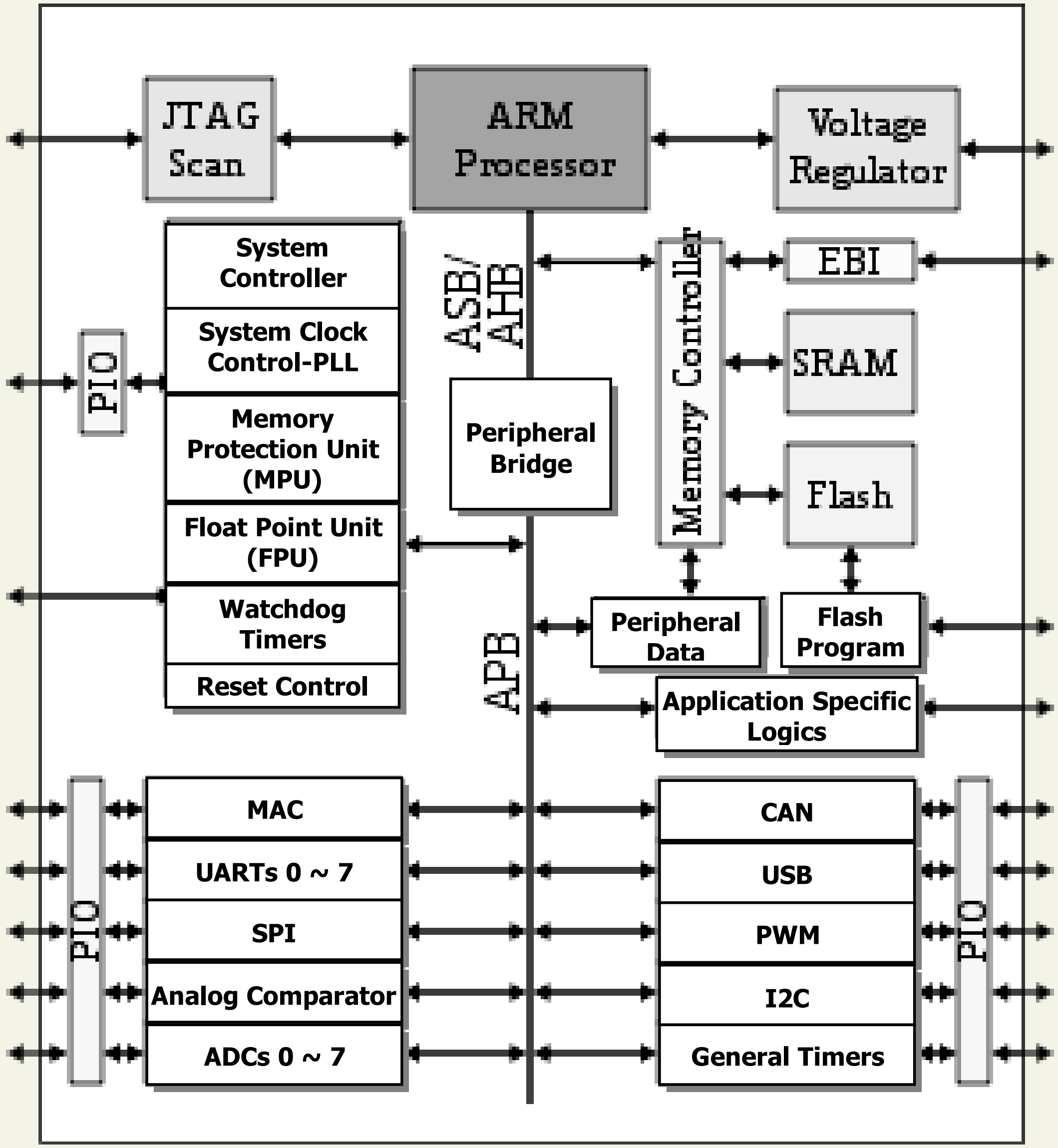


Figure 2.1 The block diagram for a general MCU.

Activity 1: GPIO Port and Pin Usage

1. List the MCU peripheral units used in Labs 1-9
2. List all GPIO ports and pins used in Labs 1-9

Note: Consider ports/pins used in “library” functions such as open_interface.c, lcd.c. For example:

- Port C
  - PC4: UART4 Rx (OI)
  - PC5: UART4 Tx (OI)

Activity 1 Resource

See last page of file Cybot-Baseboard-LCD-Schematic.pdf

LP: LaunchPad board

OI: Open Interface (iRobot)

LCD: Display

PB: Push Button

SW: Switch

Only some of these functions and pins are used in CPRE 288 labs.

Review lab code as needed.

GPIO Port A	Function	288 Inputs/Outputs
Pin 7:	LP_I2C1_SCL / IMU SCL	
Pin 6:	LP_I2C1_SDA / IMU SDA	
Pin 5:	SSIO CLK	
Pin 4:	SSIO Fss	
Pin 3:	SSIO Rx	
Pin 2:	SSIO Tx	
Pin 1:	UART0 Tx	
Pin 0:	UART0 Rx	
GPIO Port B		
Pin 7:	IMU I2C ADDR / GPIO 1	T0CCP1
Pin 6:	IMU I2C RSTN / GPIO 2	M0PWM0
Pin 5:	SERVO_2	
Pin 4:	IR1	AIN10
Pin 3:	PING_CAP_1	T3CCP1
Pin 2:	IMU I2C INT / GPIO 3	
Pin 1:	LP_UART1_Tx	
Pin 0:	LP_UART1_Rx	
GPIO Port C		
Pin 7:		
Pin 6:		
Pin 5:	OI_UART4_Tx	
Pin 4:	OI_UART4_Rx	
Pin 3:	JTAG TCK/SWCLK	
Pin 2:	JTAG TMS/SWDIO	
Pin 1:	JTAG TDI	
Pin 0:	JTAG TDO/SWO	
GPIO Port D		
Pin 7:	ADDR_JMP_4	
Pin 6:	LCD_RW	
Pin 3:	LCD_RS	
Pin 2:	LCD_EN	
Pin 1:	Connected to other ports through R9 and R10	
Pin 0:		
GPIO Port E	PUSH BUTTONS	
Pin 5:	AIN8	
Pin 4:	AIN9	
Pin 3:	PB_SW4	
Pin 2:	PB_SW3	
Pin 1:	PB_SW2	
Pin 0:	PB_SW1	
GPIO Port F		
Pin 4:	LCD DATA7	
Pin 3:	LCD DATA6	
Pin 2:	LCD DATA5	
Pin 1:	LCD DATA4	
Pin 0:	USR_SW2	OI shutoff

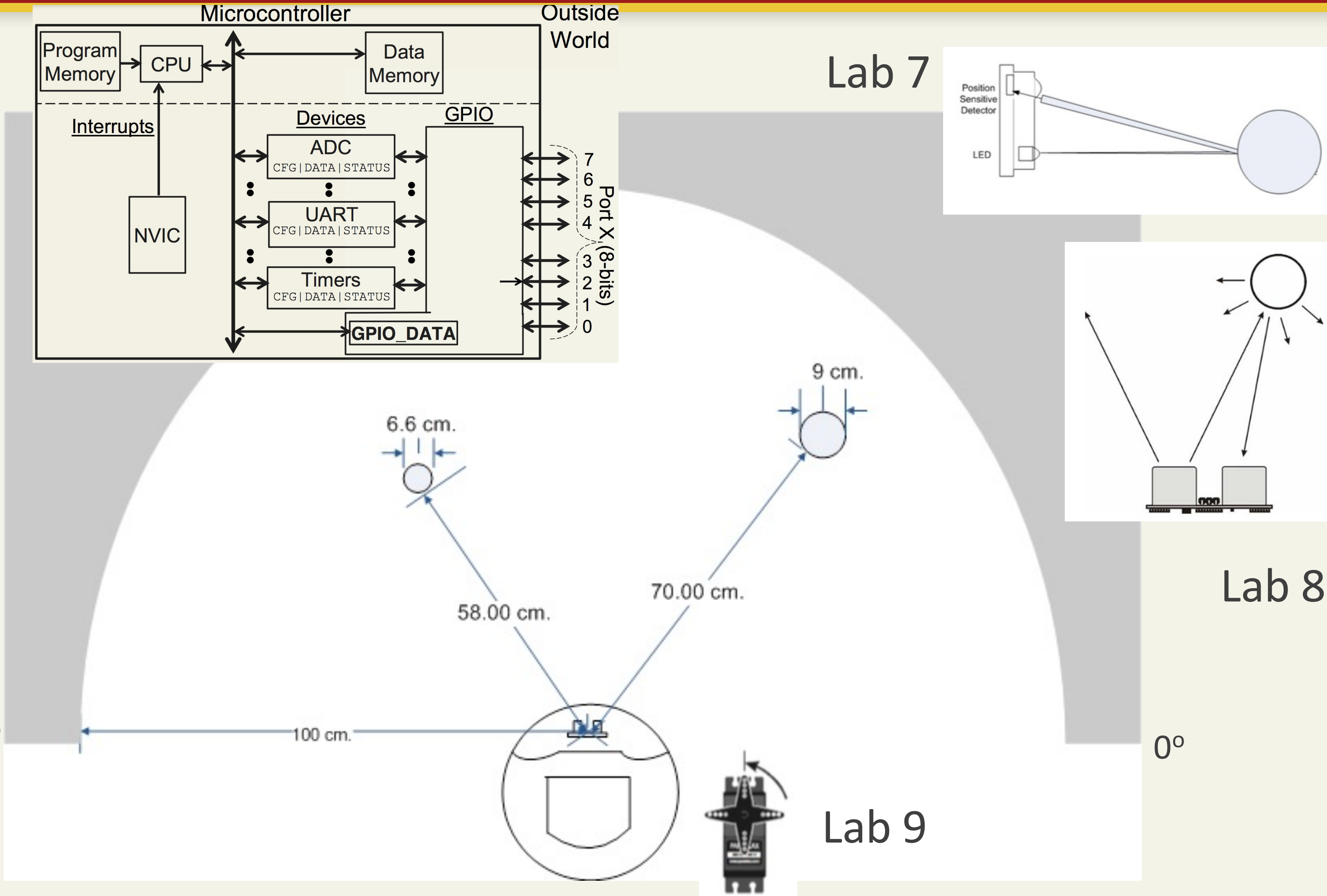
RED and CYAN labels are reserved pins, okay to ignore for this activity.

Object Detection and Analysis

Lab 7

Lab 8

180°

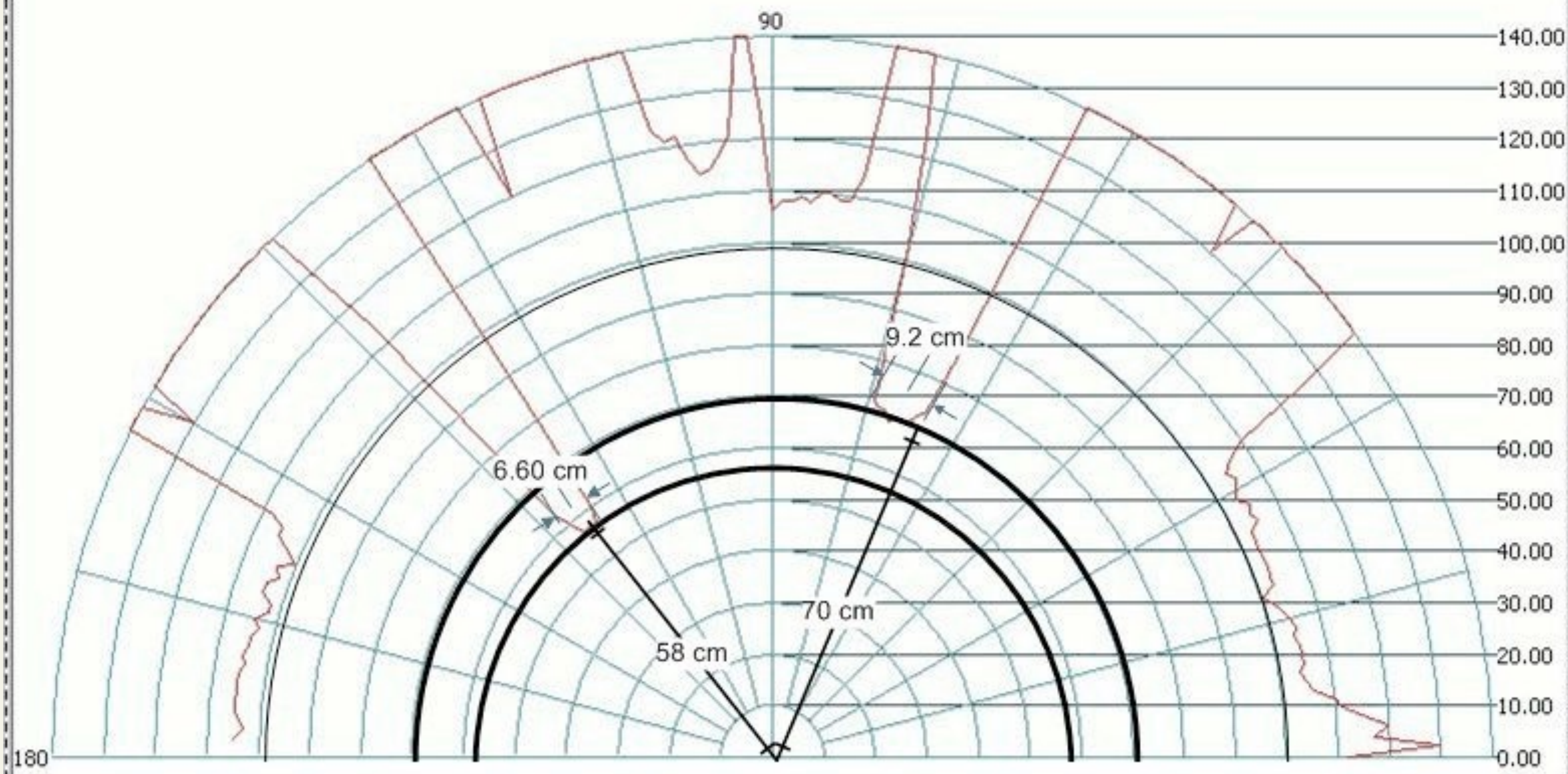


0°

Lab 9

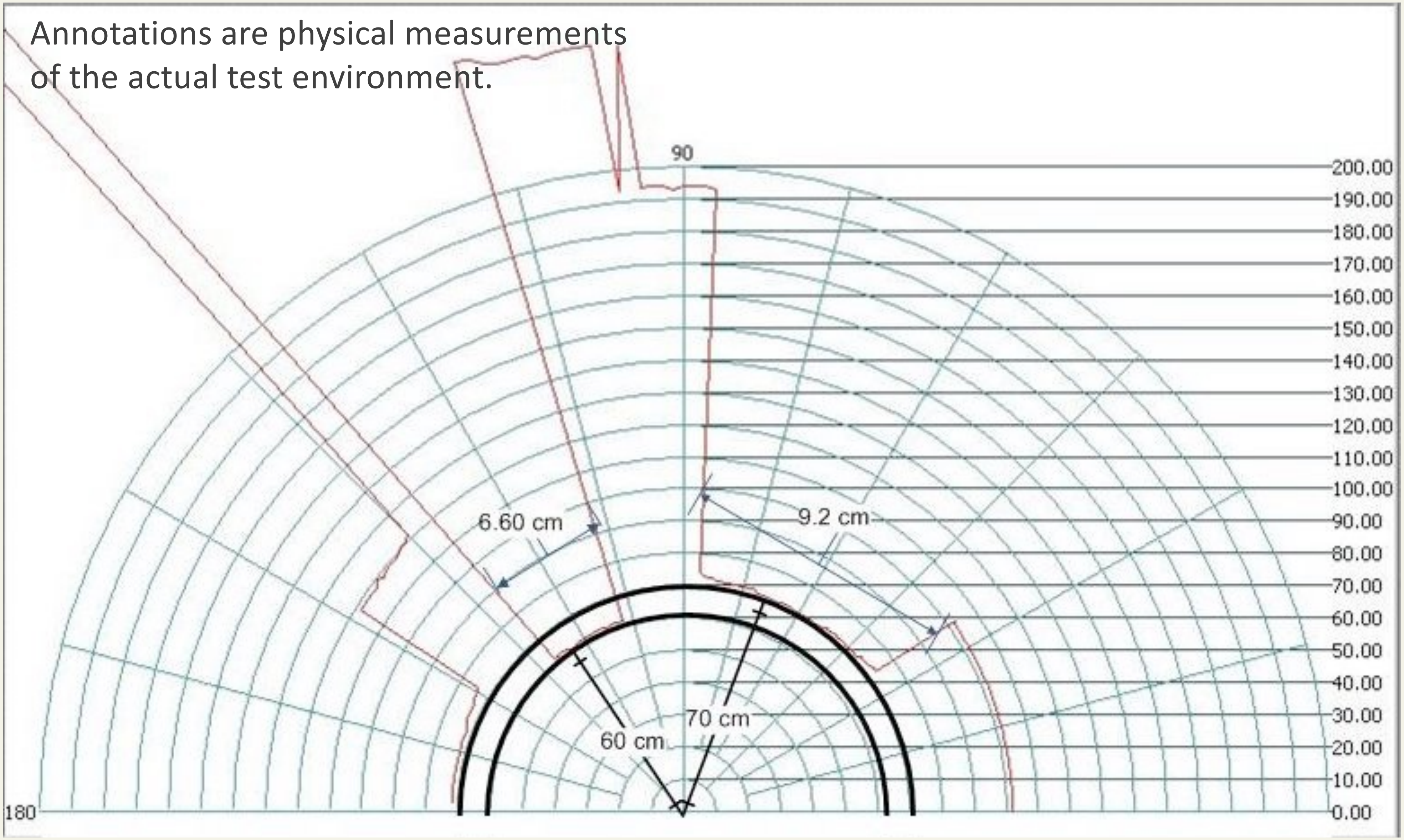
Results from IR Sensor Sweep

Annotations are physical measurements of the actual test environment.



Results from PING))) Sensor Sweep

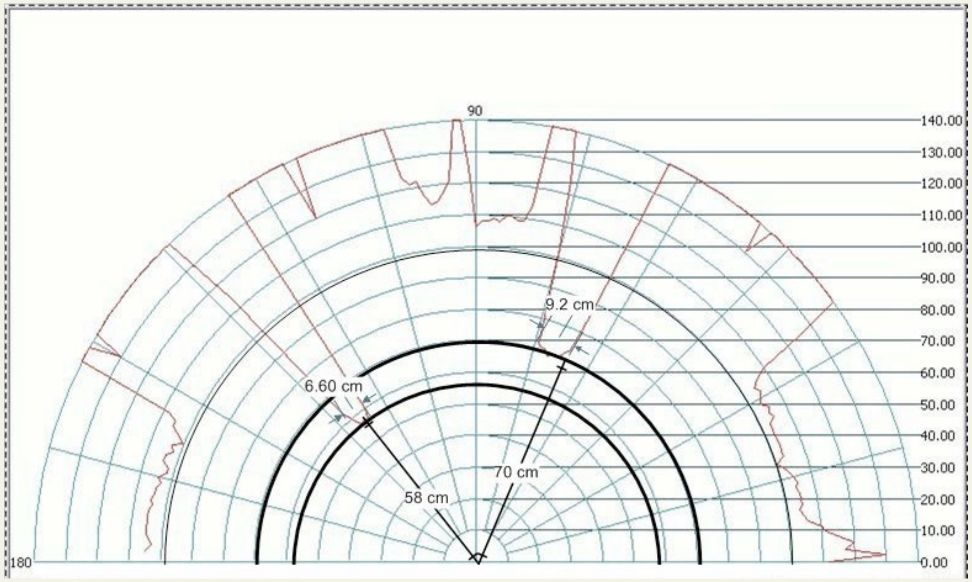
Annotations are physical measurements of the actual test environment.



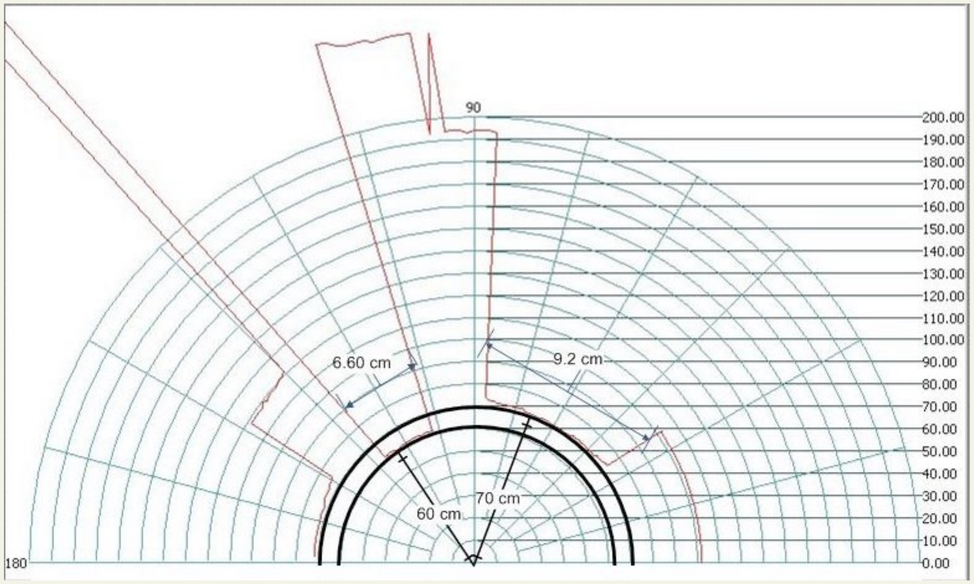
Activity 2: Valid Sensor Data

1. What is the operational range (in cm) of each sensor?
2. Identify regions of valid sensor data that fall within the range of the sensor.

IR Sensor



PING))) Sensor



Activity 3: Estimating *w* from IR Sensor Plot

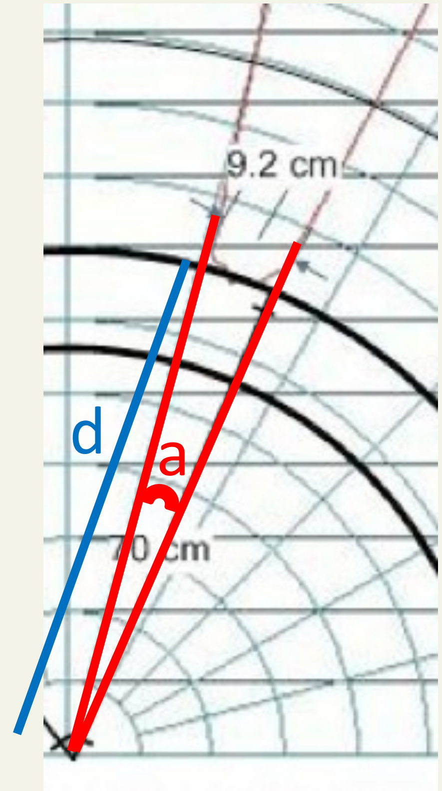
Refer to the IR sensor data plot.

Let *d* be the distance from the sensor to the object.

- The actual *d* is shown as configured in the test environment. Use that value for this activity.

Let *a* be the angle between the edges of the objects.

- The angle *a* will be calculated in your program using sensor data. Estimate it for this activity.

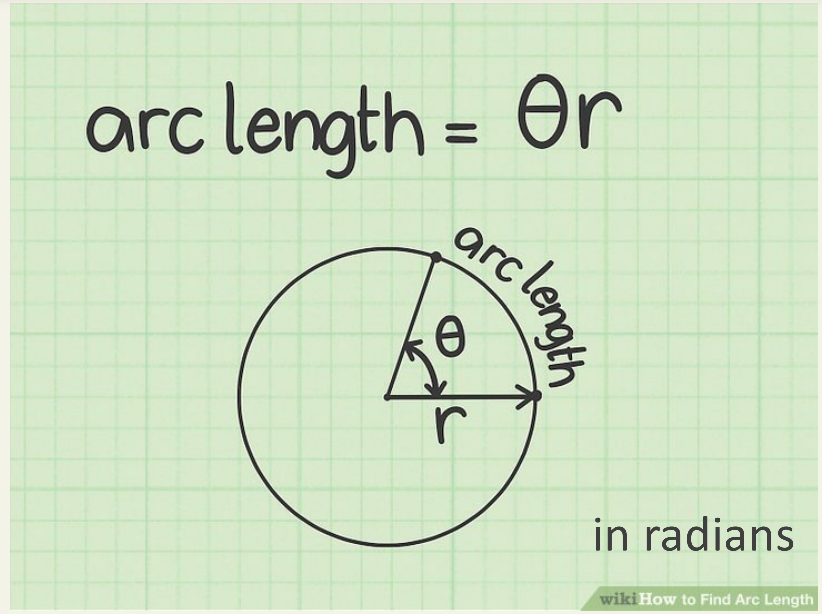


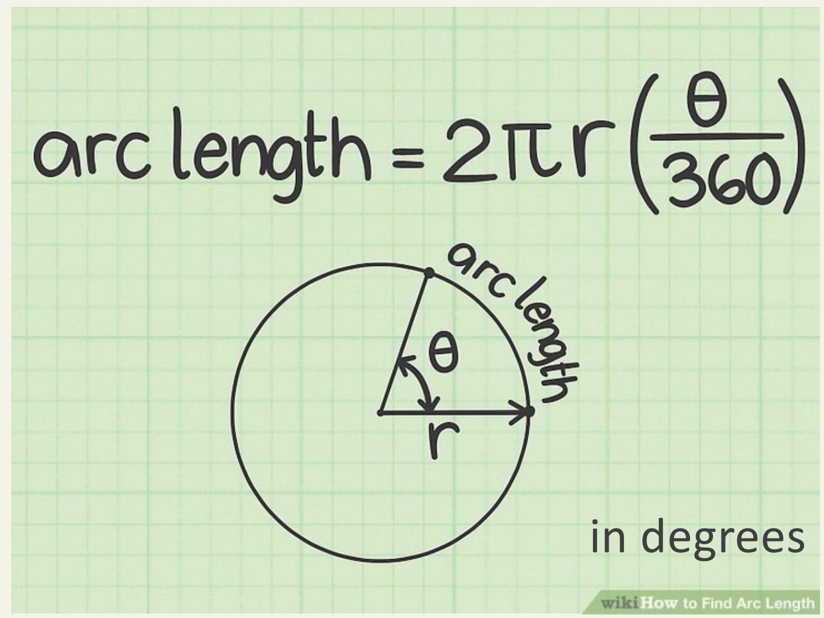
1. Using the plot, estimate the angle *a*.
2. Using *d*, *a* and geometry knowledge, estimate the width *w* of the object.

Activity 3 Resource

https://www.wikihow.com/Find-Arc-Length

An arc is any portion of the circumference of a circle. Arc length is the distance from one endpoint of the arc to the other. Finding an arc length requires knowing a bit about the geometry of a circle. Since the arc is a portion of the circumference, if you know what portion of 360 degrees the arc's central angle is, you can easily find the length of the arc.



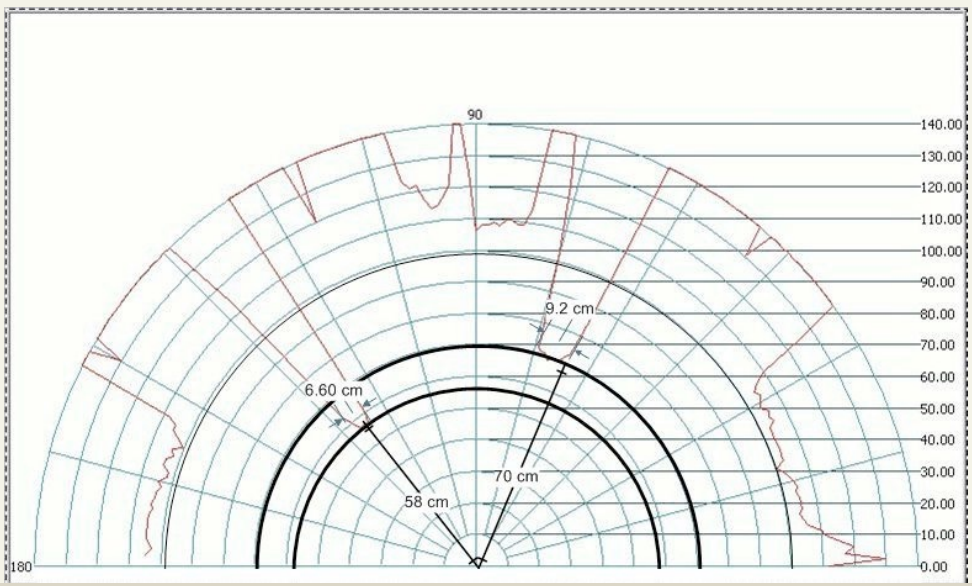


Activity 4: Calculating *a* and *d* in Code

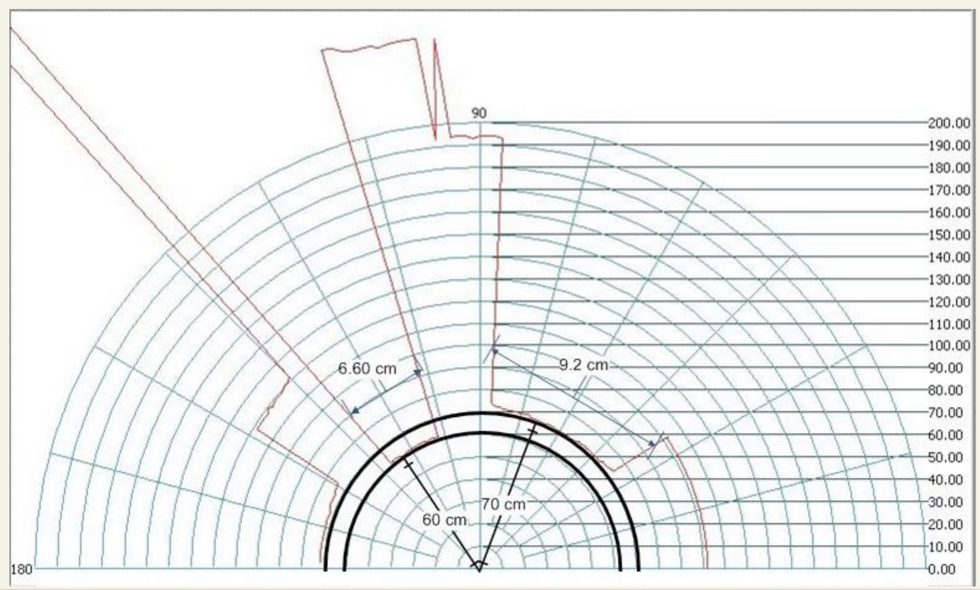
1. How is “a” determined in your program?
2. How is “d” determined in your program?

Hint: Start with calculating *a* using IR sensor data. Then combine that with PING))) sensor data to calculate *d*.

IR Sensor



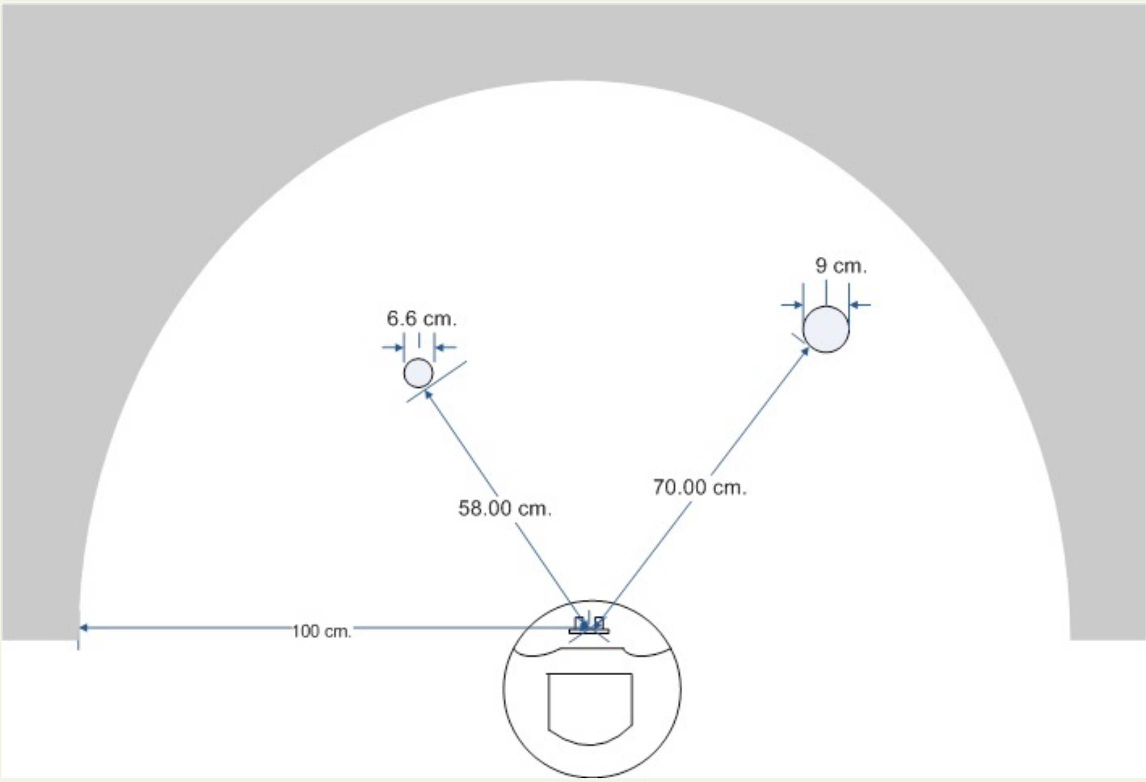
PING))) Sensor



Activity 5

Think about the actual environment and potentially problematic configurations or issues.

For example, what if objects are near each other?



Other Questions to Think About

- Why does accurate degree measurement using the servo (done through calibration) help with accurate object detection?
- Why does accurate distance measurement using the sensors (done through calibration) help with accurate object detection?

Object Detection and Analysis in the Project

Data analysis is essential in order to use the sensor data to make decisions. In the project, you will need to collect data and then analyze data to determine the number and type of objects in front of the CyBot. The IR sensor and PING))) sensor labs provide you with distance measurements in centimeters that you will use. Object detection is done by distinguishing between valid distance measurements pertaining to objects, in contrast to background noise.

Data Collection Task

Sweep both the PING))) sensor and the IR sensor over the 180-degree range of the servo, taking distance measurements from both sensors every 2 degrees and transmitting the data to PuTTY. In the test environment, objects being scanned should be 35 to 80 cm from the robot. Objects should be placed far enough apart to provide a measurable gap between them. An example of the output sent to PuTTY:

```
Degrees        IR Distance (cm)      Sonar Distance (cm)

0              120                   324
2              123                   330
4              119                   363
6              40                    40
8              40                    40
10             40                    41
...
```

Data Analysis Task

Determine the location of at least one object found by the sensors and estimate object width. To detect objects for the purpose of navigation, we are interested in finding:

1) Distance to the object
2) Angular location of the object
3) Radial size of the object (the number of degrees within a sweep in which the object appears)
4) Linear width (the actual width of the object)

These features are interrelated in data analysis due to the nature of the data collected as shown in the plots.

Data Analysis Task (continued)

Distance measurements are needed to detect the presence of an object, and from these measurements, the edges of the object can be estimated at particular angular positions. These positions can help estimate the center of the object for more accurate distance, location and width measurements.

The distance, location and width of an object should be displayed on the LCD and PuTTY.

Data Analysis Task (continued)

Start by defining the size of the object as its radial size (the number of degrees within the sweep in which the object appears). For example, if the sensors start seeing an object at servo angle 30 degrees and stop seeing it at 35 degrees, the radial size is 5 degrees.

You will also estimate the actual linear width of the object. The actual linear width is related to the radial width and can be found using geometry or trigonometry principles. This is important since objects with the same radial width value may have different linear widths (e.g., a bigger object farther away than a smaller object).

Other Tasks

Detect multiple objects, count the number of objects, and identify the smallest object

While performing the sweep of the sensors, assign an index to each object (1 for the first object encountered, 2 for the next, etc.). The index, distance, location and width of the objects should be displayed on PuTTY, and information for the smallest object should be displayed on the LCD and PuTTY.

Use user-assisted calibration for accurate object position and width measurements

Start with a calibration phase in which the user is interacting with the program. The calibration phase will result in a conversion factor that computes accurate positions. One approach to consider is having the user provide the position, distance and width of an object as part of the calibration.