Module-05: Ball beam balance system

In the previous modules, you were introduced to different aspects of designing an automated robotic system. In Module 01 you were introduced to programming the Arduino Uno. Arduino is at the heart of all the systems that you have designed in the lab so far. It is the device you have been programming to act as an controller. In Module 02 you were introduced to the idea of sensor placement and calibration. You might have observed that for the dino to properly evade the obstacles, you had to place the LDRs in the correct positions. Further, you also had to calibrate the time between sensing the obstacles and jumping over the obstacles as the speed of the obstacles arriving increased over time. In Module 03 you have programmed the Arduino to act as an ON-OFF controller and in Module 04 you have used the Arduino as a PID controller. In the next module, we are introducing another important facet of designing an automated system. In all the previous modules, you have not physically designed a system. Mechanical design is a crucial part of a robotic system. Hence, this module is divided into two parts: (i) the physical design of the system, and (ii) design of the controller using an Arduino to achieve the goals.

Objectives: A ball is placed on a beam, see Figure 1, where it is allowed to roll with 1 degree of freedom along the length of the beam. A lever arm is attached to the beam at one

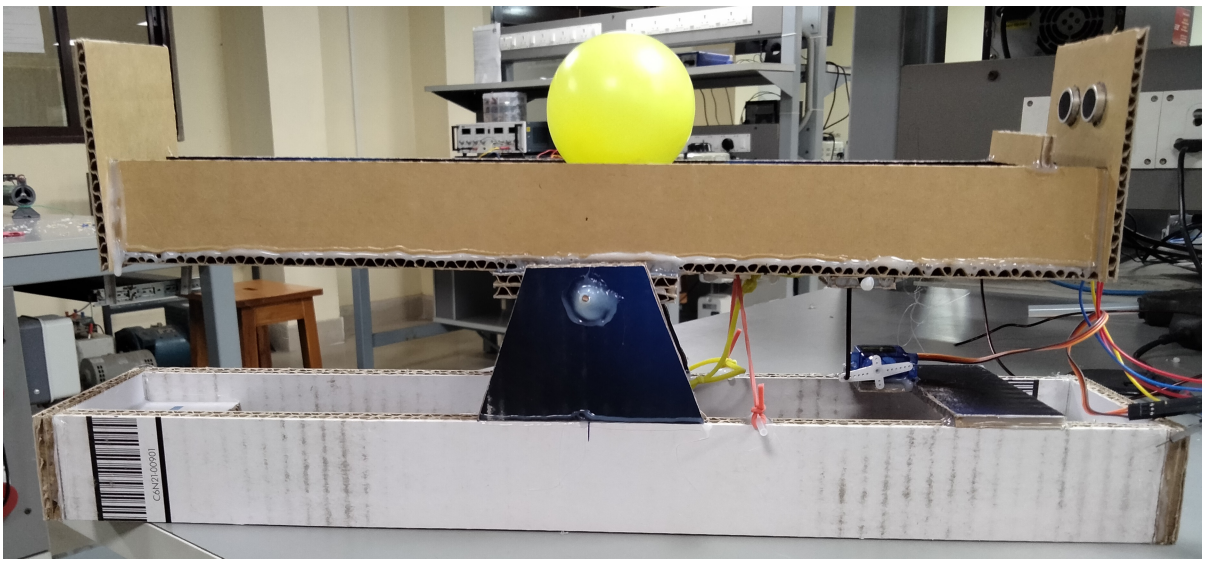

Figure 1: Front view of ball-balance system (balanced position)

end and a servo gear at the other. As the servo gear turns by an angle θ, the lever changes the angle of the beam by α. When the angle is changed from the horizontal position, gravity causes the ball to roll along the beam. You need to design a controller so that the ball's position can

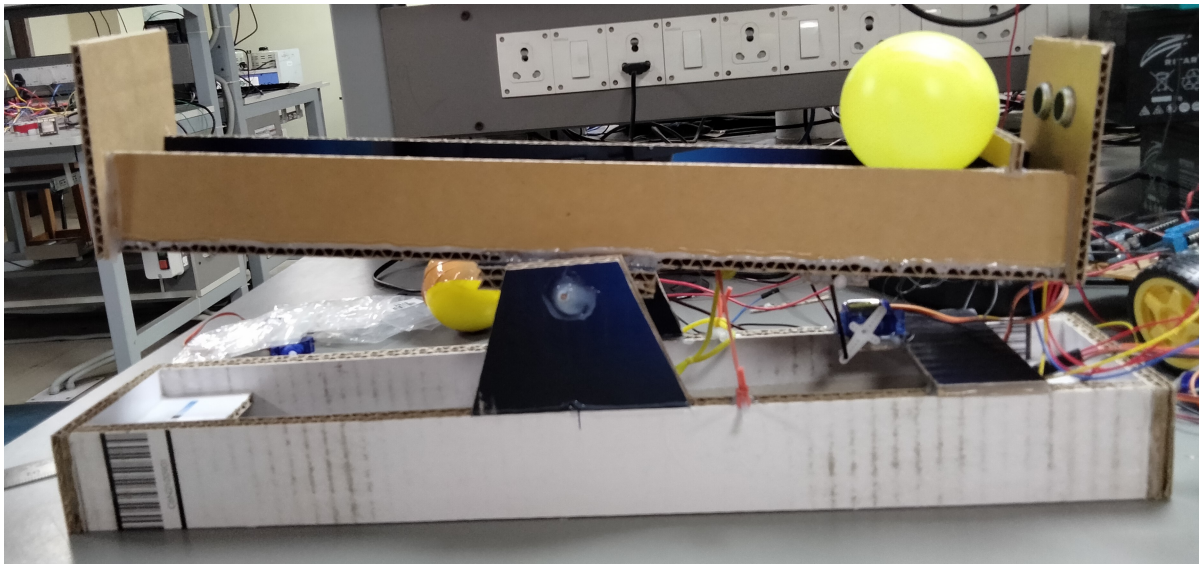

Figure 2: Front view of ball-balance system (unbalanced position)

be manipulated and the beam is balanced (see video in File section of Teams).

Physical setup: In order to make the physical ball beam system, you will be using cardboard. Cardboards should be glued together using the glue gun provided to you. The dimensions of the ball beam set-up in Figure 1 are as given in the Figure 3 below:

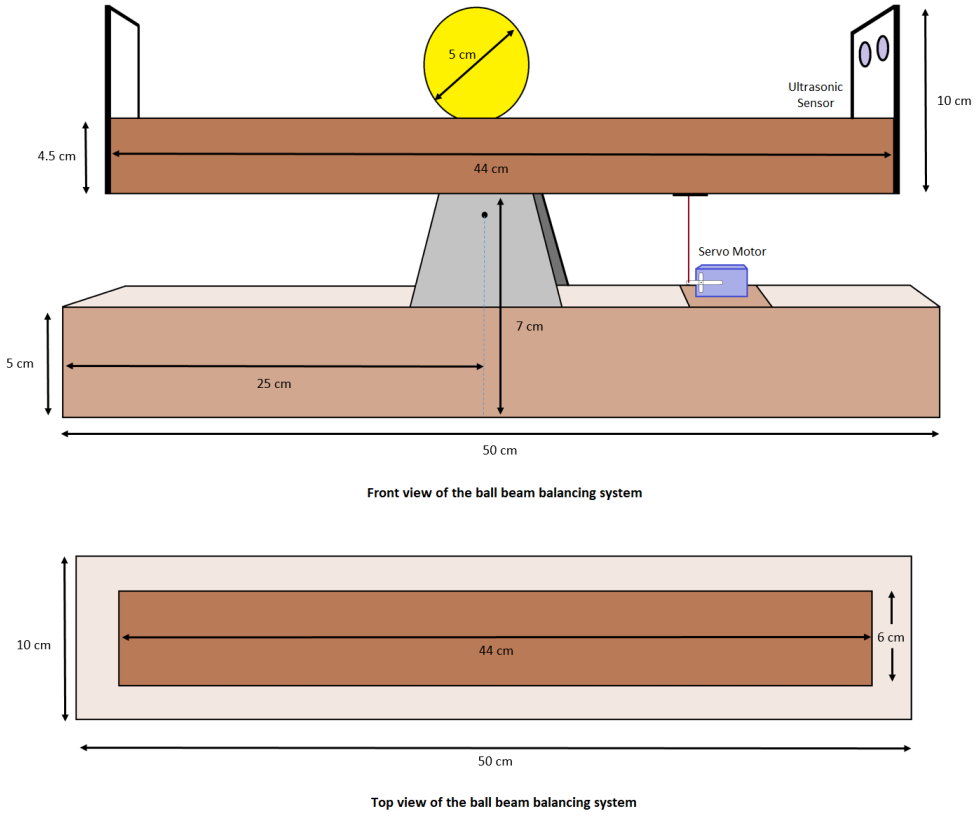

Figure 3: Dimensions of ball-balance system

The dimensions provided in the figure is for reference only. You can design your own ball beam system; however keep the following points in consideration while designing the system.

1. The length of the beam on which the ball is resting should be between 30 cm to 45 cm. The width of top beam should be less than the width of the lower beam. This will ensure that that the top beam does not touch the bottom beam when it is at an inclined position.

2. Place the servo-motor such a way that the shaft connected to the top beam is almost at right-angles to ground when the beam is at balanced position. This is visible in Figure 1.

3. The position of the ultra-sonic sensors should be such that the transmitted waves from the sensor hits the center of the ball. Wrong sensor placement will lead to erroneous readings which will make it difficult for you to tune the controller parameters.

4. The top beam should be fixed to the spindle firmly. Ensure that the beam does not move horizontally. Any horizontal motion of the beam will introduce unnecessary vibrations into the system which will again make it difficult to tune the controller parameters.

Software setup: Once the hardware setup is completed, you need to start designing the controller using Arduino Uno. The following points need to be taken care of while writing the code.

1. The error signal to be feed to the controller is the difference between the reference point and the sensor data. The reference point here is the position of the ball when the beam is balanced. Hence you need to find this value first. The data from the ultrasonic sensor will provide you the instantaneous position of the ball. This position subtracted from the reference position will give you the error signal.

2. The data received from the sensor might not be steady. Hence you need to use a filter to smoothen measurements. You can use moving average filter to achieve this. Kalman filter is also an option.

3. Start with the P controller first and tune the set-up. Use the serial port plotter to check whether you will need I or D controller and accordingly tune K_I and K_D gains of the PID controller.

Report Guidelines: The report to be submitted must have the following:

1. Objectives of the module.

2. Circuit diagram with proper labelling.

3. Block diagram of the system designed.

4. A picture of the entire setup.

5. A working video of the setup will have to be uploaded.

6. Flowchart of the algorithm.

7. Observations, if any.

    - Snap shots of the serial monitor needs to be submitted with your observations. You have to show how is the performance of the system effected for different values of K_p, K_I, and K_d.
    - Plot of the output from the ultrasonic sensor with and without the filter.

8. Answers to the questions below.

Answer the following:

1. Draw the block diagram of the closed loop system you have implemented in this module. What is the set point of your system?

2. How did you tune the controller gains?

3. What is your observation when the value of K_p is increased keeping K_I and K_d zero? Can you relate it to the theory you have studied in Control Theory course?

4. What kind of filter have you used to clean the data recorded (High pass, Low pass, etc)? Based on the output plot from the sensor, why do you think the filter helped in cleaning the data?

5. Find the value of K_p at which your system becomes unstable.

6. Find the mathematical model of the system you designed. Can you relate why the system becomes unstable at high K_p values?