Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

Task 3

Balance ball on platform

[*Last Updated on:* ***25th November 2020, 22:30 Hrs***]

- 1. Problem Statement
- 2. Given
  - 1. CoppeliaSim's Scene File (`task_3_scene.ttt`)
  - 2. Main Script (`task_3.py`)
  - 3. Test executable (`test_task_3.exe` or `test_task_3`)
- 3. Getting Started
- 4. Understanding the Task
  - Ball balancing platform designed by team in Task 1C
  - CoppeliaSim Scene
  - Python Script
    1. `init_setup(rec_client_id)`
    2. `control_logic(center_x,center_y)`
    3. `change_setpoint()`
- 5. Testing the Solution
  - 1. Using `task_3.py`
    2. Test using `test_task_3.exe` or `test_task_3`
- Submission Instructions

The aim of this task is as follows:

- Develop a **control logic** to **balance the ball on top of the ball balancing platform**.
- Apply the concepts **learnt in Task 1, Task 2** and build a **python remote API file** to stabilize the ball at a particular setpoint on the table using **vision sensor's image** in the given CoppeliaSim scene.

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

NOTE: Before proceeding further, make sure you have thoroughly understood the concepts of **Task 1** and **Task 2**.

1. Problem Statement

Develop an algorithm **to balance a ball at a particular setpoint** on the platform in the given CoppeliaSim scene.

2. Given

1. CoppeliaSim's Scene File (`task_3_scene.ttt`)

- A scene file i.e. `task_3_scene.ttt` of CoppeliaSim software as shown in Figure 1.

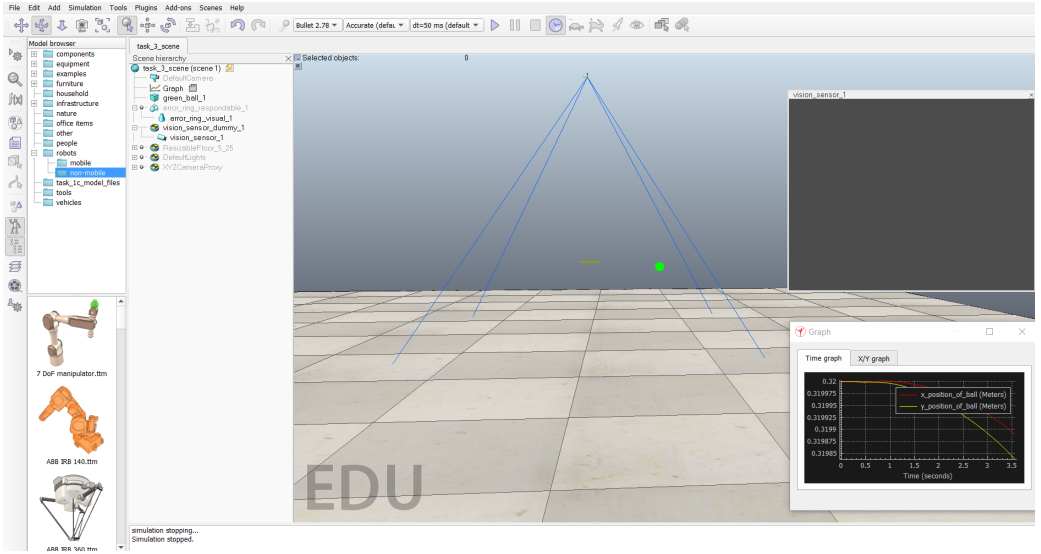


Figure 1: CoppeliaSim scene file (task_3_scene.ttt).

- The **objects in the scene** along **with their names and uses** are given in Table 1.

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

Objects	Name in the scene	Use(s)
Green Ball	*green_ball_1*	**Green color ball** that is supposed to be balanced.
Error Ring Respondable	*error_ring_respondable_1*	Representing **respondable part** of the **error ring**. It will visually let the team know whether their ball is in the required error range or not.
Error Ring Visual	*error_ring_visual_1*	Representing **visual part** of the **error ring**. It will visually let the team know whether their ball is in the required error range or not.
Vision Sensor Dummy	*vision_sensor_dummy_1*	**Dummy** used to position and orient Vision Sensor.
Vision Sensor	*vision_sensor_1*	Works as the **camera** in the scene. The **output of vision sensor** is supposed to be used for **Image Processing**. It has resolution of 1024x1024 pixels.
Graph	*graph_1*	**Plots** the **x and y CoppeliaSim co-ordinates of the ball** when the simulation is running.

Table 1: Objects along with their names and uses in the given scene file.

- A floating view of the output of ***vision_sensor_1*** is also shown.

NOTE:

- In this task you are **NOT** allowed to **remove** the above mentioned objects.
- Any change in the **names of the objects, their parent child relationships, their properties, position, orientation** etc. will result in **poor evaluation** and hence low marks.

- Teams will have to **import their ball balancing platform** designed in Task 1C.

2. Main Script (`task_3.py`)

- The python script i.e. `task_3.py` **NEEDS** to be edited by the respective teams.
- It **contains functions** which are going to be called by the `test_task_3.exe` or `test_task_3`.
- The **details of each function** is mentioned in the file and in this documentation. Read **both the files carefully** before attempting the task.

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

3. Test executable (`test_task_3.exe` or `test_task_3`)

- The team **should run this executable ONLY after they have completed** writing the `task_3.py` script.

---

3. Getting Started

- Download the following zip file containing the above mentioned files. **Right-click** on the hyperlink and select **Save Link As...** option to download.
  - **Windows OS Users**:
    - **task_3_balance_ball_on_platform_windows.zip**
  - **Ubuntu OS Users**:
    - **task_3_balance_ball_on_platform_ubuntu.zip**
  - **Macintosh OS Users**:
    - **task_3_balance_ball_on_platform_macintosh.zip**

NOTE: The browser might warn that the file can harm your PC, but it will not and you can safely download it.

- **Extract** the downloaded zip file.
- Copy `task_2a.py` , `task_1a_part1.py` and `task_1b.py` files made by your team in **Task 2A**, **Task 1A** and **Task 1B** respectively.
- Paste them in the extracted folder.
- Make sure you add following files in the same directory where all the Python scripts of this **Task 3** are present.
  - `sim.py`
  - `simConst.py`
  - `remoteApi.dll` (***for Windows***) OR `remoteApi.so` (***for Linux***) OR `remoteApi.dylib` (***for Macintosh***)
- Refer **Task 0 Test Setup** for further details.

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

- **After completing the above steps**, your folder should have the following files:
    - `task_3_scene.ttt`
    - `test_task_3.exe` / `test_task_3`
    - `task_3.py`
    - `task_2a.py`
    - `task_1a_part1.py`
    - `task_1b.py`
    - `sim.py`
    - `simConst.py`
    - `remoteApi.dll` (***for Windows***) OR `remoteApi.so` (***for Linux***) OR `remoteApi.dylib` (***for Macintosh***)

Now, read the following instructions carefully. **Any deviation from the listed instructions will result in poor evaluation and hence low marks**.

---

4. Understanding the Task

Ball balancing platform designed by team in Task 1C

- Make sure you have gone through Tips and Tricks to improve your Ball Balancing Platform Design document.
- For this task, you will have to **check all the boxes of local respondable mask** for ***base_plate*** and **objects directly connected to the *base_plate***. These are as follows:
    - ***base_plate_respondable***
    - ***yoke_respondable*** and
    - ***servo_holder_respondable***

> **NOTE:** This is **extremely important** since it is **NOT** allowed to put objects into each other.

- Refer the gif in Figure 2 to learn how to turn on the local respondable mask for an object in CoppeliaSim scene.

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

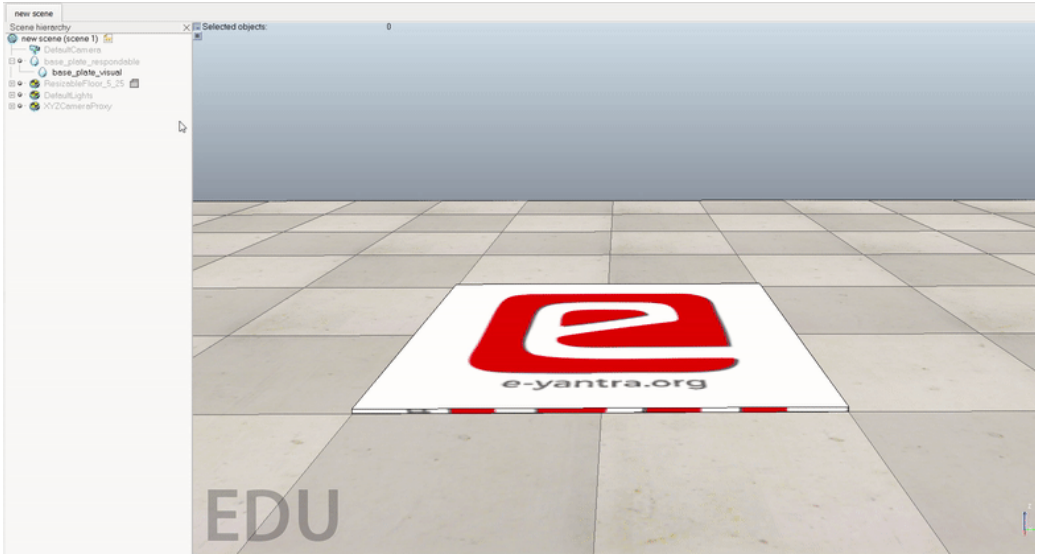


Figure 2: Enable local respondable mask for an object in CoppeliaSim scene.

- In order to **improve the processing power and reduce the lag** between python script and CoppeliaSim scene, **enable** the **renderable** option only for:
  - *top_plate_respondable* and
  - *top_plate_visual*
- Only **renderable objects** are **seen by the vision sensor**.
- To learn more about **renderable objects** click here.
- Refer the gif in Figure 3 to learn how to make an object **renderable** in CoppeliaSim scene.

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

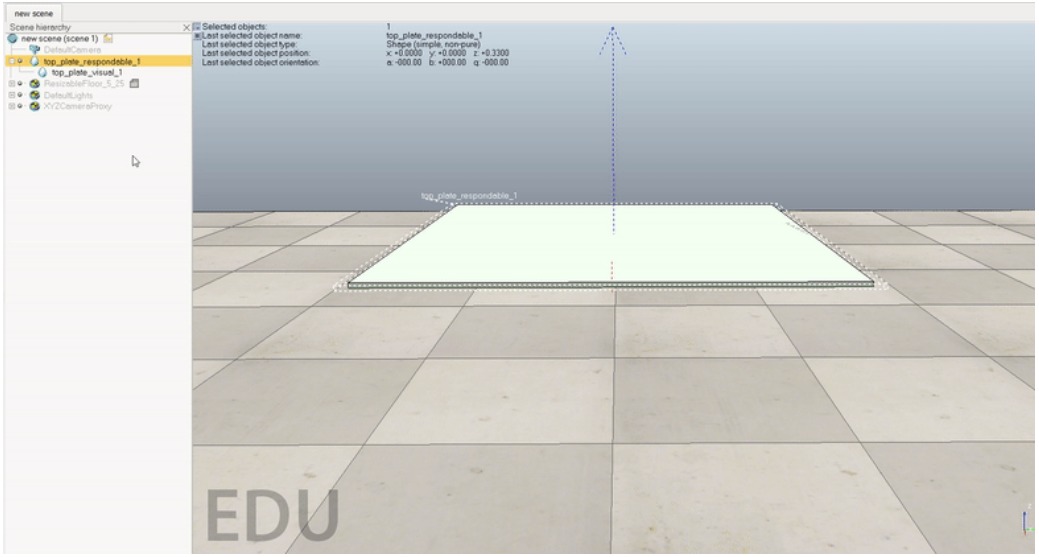


Figure 3: Make an object renderable in CoppeliaSim scene.

- Click on the **play button** of simulation to **ensure everything is working correctly**. You shou get the **same output as task 1c**.
- If there are **vibrations/oscillations in the design**, it means **some object has coincided with another**.
- **Export your model** by referring Figure 10 of the Task 1C - Design Ball Balance Platform document.

CoppeliaSim Scene

- After you open the scene, you have to **add the model file**.
- Paste the **.ttm** file in:
  - Ubuntu OS - `CoppeliaSim_Edu_V4_0_0_Ubuntu18_04/models/`
  - Windows OS - `C:\Program Files\CoppeliaRobotics\CoppeliaSimEdu\models\`
- Refer Figure 2 of Task 1C - Design Ball Balance Platform to understand better.
- Position the parent of your design at **[0, 0, 0.0055]** and orient at **[0, -90, 0]** with respect to th world frame.
- **To check** that you have **positioned and oriented your design correctly**, make sure that whe you hit the play button of simulation, you are able to **match** the **position of the ball** in the

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

floating view of *vision_sensor_1* and your **design**.

NOTE:

- Make sure that the **height** of your design or the **z co-ordinate** of the ***top_plate_respondable*** is less than or equal to **0.35** in CoppeliaSim co-ordinates.

- Now you need to configure **ONLY** the **required revolute joints** (depends on your logic and design) as a motor.
- Refer the gif in Figure 4 to check the **Motor enabled** and **Control loop enabled** option for the required revolute joints.

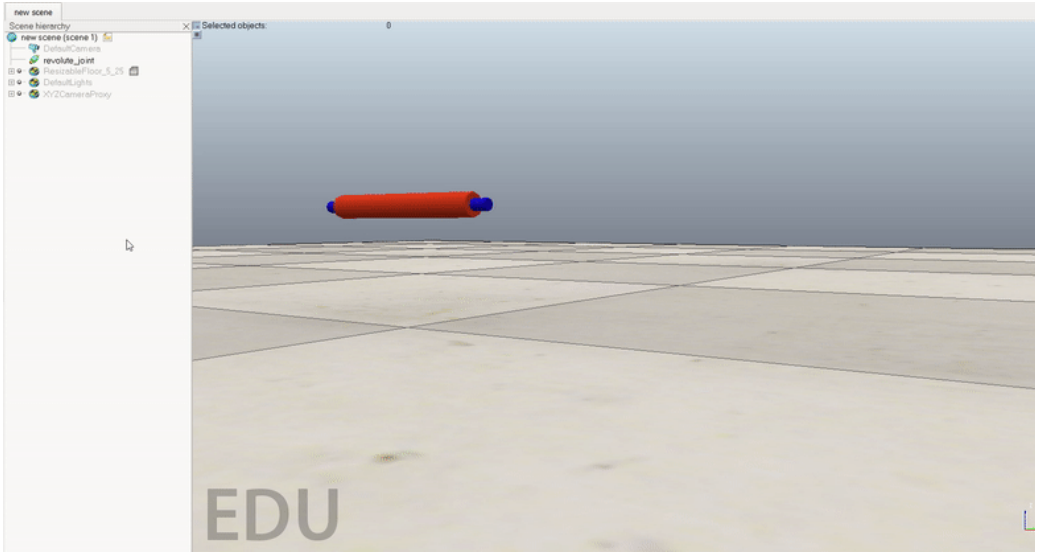


Figure 4: Check the Motor enabled and Control loop enabled option for the required revolute joints.

- Make sure that the parameters in **Joint Dynamic Properties** dialog box are as shown in Figur 5.

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

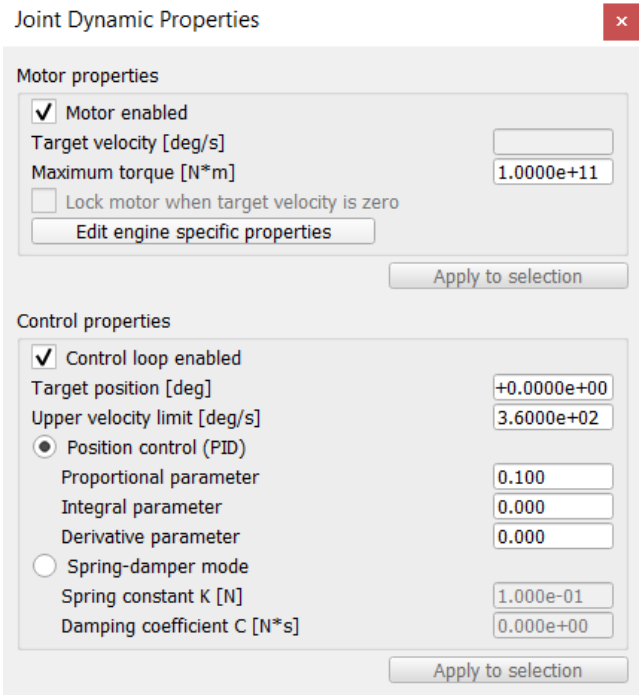


Figure 5: Properties of Motor enabled and Control loop enabled option in Joint Dynamic Properties dialog box for the required revolute joints.

- In order to **understand more about joints** in CoppeliaSim click here. This will **help you to determine which revolute joints** are supposed **to be used as a motor**.
- To learn about the working of a simple servo motor watch the below video by **How to Mechatronics** (You can skip the interfacing with Arduino part for now).

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

How Servo Motors Work & How To Control Servos using Arduino

NOTE:

- Since the servo motor has its **own angular control**, the code **needs to provide it with the correct angle**.
- This angle will be **generated by your control logic** and **communicated to** the **servo motor** in CoppeliaSim using **Python Legacy Remote API**.

Python Script

- The given script i.e. `task_3.py` has **2 functions** that are supposed to be completed by the team.
- The remaining **functions** that are called by `task_3.py` or `test_task_3.exe` / `test_task_3` ma **require minor changes**.
- These functions along with their uses are shown in **Table 2**.

Sr. No.	Name	Use	Python script to be used

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

Sr. No.	Name	Use	Python script to be used
1	`init_setup(rec_client_id)`	**Initialize** commands and/or variables required in the script.	`task_3.py`
2	`control_logic(center_x,center_y)`	**Implement the control logic** to balance the ball at a particular setpoint on the table	`task_3.py`

Table 2: Functions to be edited by teams in **task_3.py**.

NOTE: You are **allowed to use helper functions** in the allocated space of `task_3.py` .

- The **details** for each of mentioned **functions in Table 2** are shown in the underlying sections

1. `init_setup(rec_client_id)`

Parameters	Details
Purpose	This function should: 1. Get all the **required handles** from the CoppeliaSim scene and store them in global variables. 2. **Initialize the vision sensor** in 'simx_opmode_streaming' operation mode (**if required**). **NOTE:** Teams **are allowed** to choose **appropriate operation mode** depending on their code and logic.
Input Arguments	`rec_client_id` Since the `init_remote_api_server()` was written in `task_2a.py` , the **client id** returned by this function should also be available to `task_3.py` . The value is stored as a global variable for `task_3.py` .
Return	None
Example Call	`init_setup(rec_client_id)`

Table 3: ***init_setup()*** function to be edited by teams in **task_3.py**.

2. `control_logic(center_x,center_y)`

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

Parameters	Details
Purpose	1. This function should **implement the control logic** to balance the ball at a particular setpoint on the table. 2. The **orientation of the top table** should **ONLY** be **controlled by the servo motor** as we would expect in a practical scenario. 3. Hence **ONLY** the **shaft of the servo motor** or in other words the **revolute joint between servo and servo fin** should have **'motor enabled' and 'control loop enabled option'** checked. 4. This function should use the **necessary Legacy Python Remote APIs** to control the revolute joints. **Important:** Since you have designed a 2 DOF ball balancing platform, you will have run your control logic for **both X and Y axes**.
Input Arguments	`center_x` : [int] the x centroid of the ball `center_y` : [int] the y centroid of the ball
Return	None
Example Call	`control_logic(center_x,center_y)`

Table 4: ***control_logic()*** function to be edited by teams in **task_3.py**.

NOTE:

- In `task_2a.py` , the **input to the** `get_vision_sensor_image()` function was ***None***. However now it is changed to ***vision_sensor_handle***.
- Hence get the handle of *vision_sensor_1* in `init_setup()` function and **store it in a global variable**.
- We will be passing this handle to `get_vision_sensor_image()` function as and when required.
- It is **recommended** to use **grayscale image** since you do not need to identify the color in this task. It will help to **greatly reduce the lag** between CoppeliaSim and Python script. You can do so by referring the link from here.
- If you are **working with grayscale image**, pass **'None'** in the color parameter of shape dictionary.

The following function is NOT to be edited.

3. `change_setpoint()`

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

Parameters	Details
Purpose	The function updates the value of global `setpoint` list after **every 15 seconds of simulation time**. This will be **ONLY** called by `test_task_3.exe` or `test_task_3` .
Input Arguments	`new_setpoint` : [list] [x_pixel,y_pixel]
Return	None
Example Call	`change_setpoint(new_setpoint)`

Table 5: ***change_setpoint()*** function to **NOT** be edited by teams in **task_3.py**.

---

5. Testing the Solution

- Before testing the solution make sure you have **plugged in your laptop to power source** and **closed unnecessary applications** open in your PC.

> **NOTE:** The installation of all software/libraries has been tested **only** on the following **64 bit OS**:
>
> - ***Windows 7, 8 and 10***
> - ***Ubuntu 16.04 and 18.04***
> - ***macOS Big Sur v11.0.1***

- The files i.e. `task_3.py` and `test_task_3.exe` will call the following functions in the given order:

> **NOTE:** The resolution of the warped image output from `applyPerspectiveTransform(transformed_image)` should **strictly** be **1280x1280** pixels.

Sr. No.	Function Name	Python script
1	`init_remote_api_server()`	`task_2a.py`
2	`start_simulation()`	`task_2a.py`

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

Sr. No.	Function Name	Python script
3	`init_setup(client_id)`	`task_3.py`
4	`get_vision_sensor_image(vision_sensor_handle)`	`task_2a.py`
5	`transform_vision_sensor_image(vision_sensor_image, image_resolution)`	`task_2a.py`
6	`applyPerspectiveTransform(transformed_image)`	`task_1b.py`
7	`scan_image(warped_img)`	`task_1a_part1.py`
8	`control_logic(center_x,center_y)`	`task_3.py`
9	`stop_simulation()`	`task_2a.py`
10	`exit_remote_api_server()`	`task_2a.py`

Table 6: List of functions that will be called by test_3.py / test_task_3.exe in the given order.

The testing of solution is **divided into two** sub parts:

1. Using `task_3.py`

- To test your solution,
  - **Activate** your **Conda environment**, **navigate to the directory** where you have downloaded all the files and run
    - `python task_3.py`
  - Open the provided scene file `task_3_scene.ttt` in CoppeliaSim.
- The simulation will run for a **maximum of 15 seconds** of simulation time.
- The ball should traverse from its initial position **(1063, 1063)** to the centre of table **(640, 640)** within **15 seconds**.
- It will print the **detected shape, color (if any), center x and center y** returned by your `scan_image()` function.
- You can **tune the parameters of your control logic** if required. Do not forget to refer Understaning Proportional Integral Derivate (PID) Controller document.
- Refer the graph in CoppeliaSim to understand how your control logic is behaving.
- If the code runs without any errors and you are **satisfied with the output** obtained, you can proceed ahead with the **next part**.

2. Test using `test_task_3.exe` or `test_task_3`

- This executable file will call the functions from `task_3.py`, `task_2a.py`, `task_1b.py` and `task_1a_part1.py` regularly.

Welcome to NB theme!
Rulebook
Task 0
Task 1
Task 2
Task 3
Tips and Tricks to improve your Ball Balancing Platform Design
Introduction to Control Systems
Understanding Proportional Integral Derivative (PID) Controller
Balance ball on platform
Task 4
Task 5
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and GitHub
Live Session 1 - 24th October 2020
Live Session 2 - 21st November 2020
Live Session 3 - 12th December 2020
Live Session 4 - 10th January 2021
Changelog

- After **every 15 seconds of simulation time, the setpoint will be changed** using the `change_setpoint()` function. You can observe the output and tune your control logic for all the setpoints.

- To record the video of simulation, click on the  button of CoppeliaSim and set the parameters as shown in Figure 6.

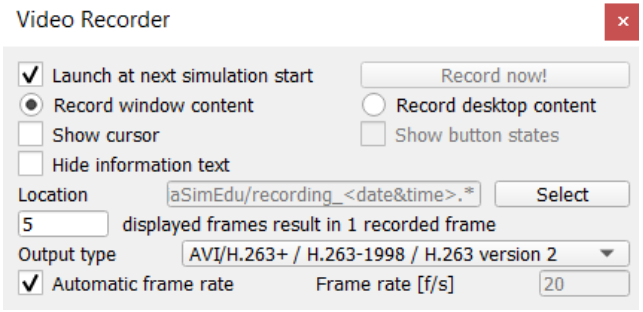


Figure 6: Video Recorder dialog box of CoppeliaSim.

NOTE:

- Do **NOT** change any of the above parameters. It may otherwise lead to **slow simulation** and **increase in lag**.
- Do **NOT** forget to **uncheck** the ***Hide information text*** option.

- Position the view of the scene as shown in Figure 7.

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

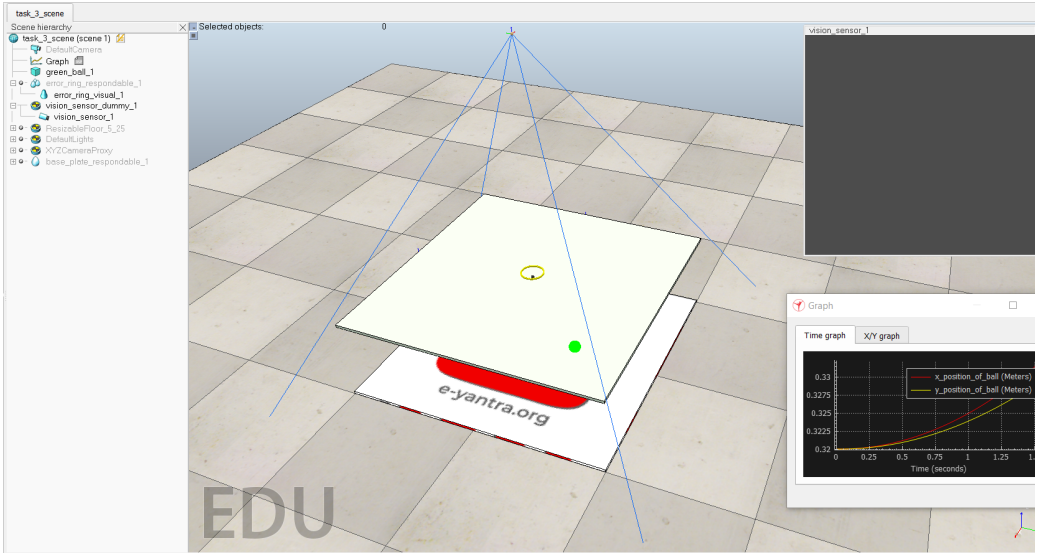


Figure 7: View of the CoppeliaSim scene while recording the video.

NOTE: In Figure 7, objects such as servo, servo holders etc. are hidden. You are **NOT** allowed to **hide** any object/joint/force sensor/dummy.

- Make sure that your **Conda environment is activated**.
- Open the provided scene file `task_3_scene.ttt` in CoppeliaSim.
- When the `test_task_3.exe` or `test_task_3` is running, make sure you **do not disturb the code or the CoppeliaSim scene**.
- Navigate to the directory **where all the files mentioned were downloaded and extracted**.

NOTE: After you run the following command(s) in your respective OS, it may take **up to 1 minute** for the file to initialize.

  - Type the following command:
    - `test_task_3.exe`
  - If there are no errors you will see the output resemble Figure 8.

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

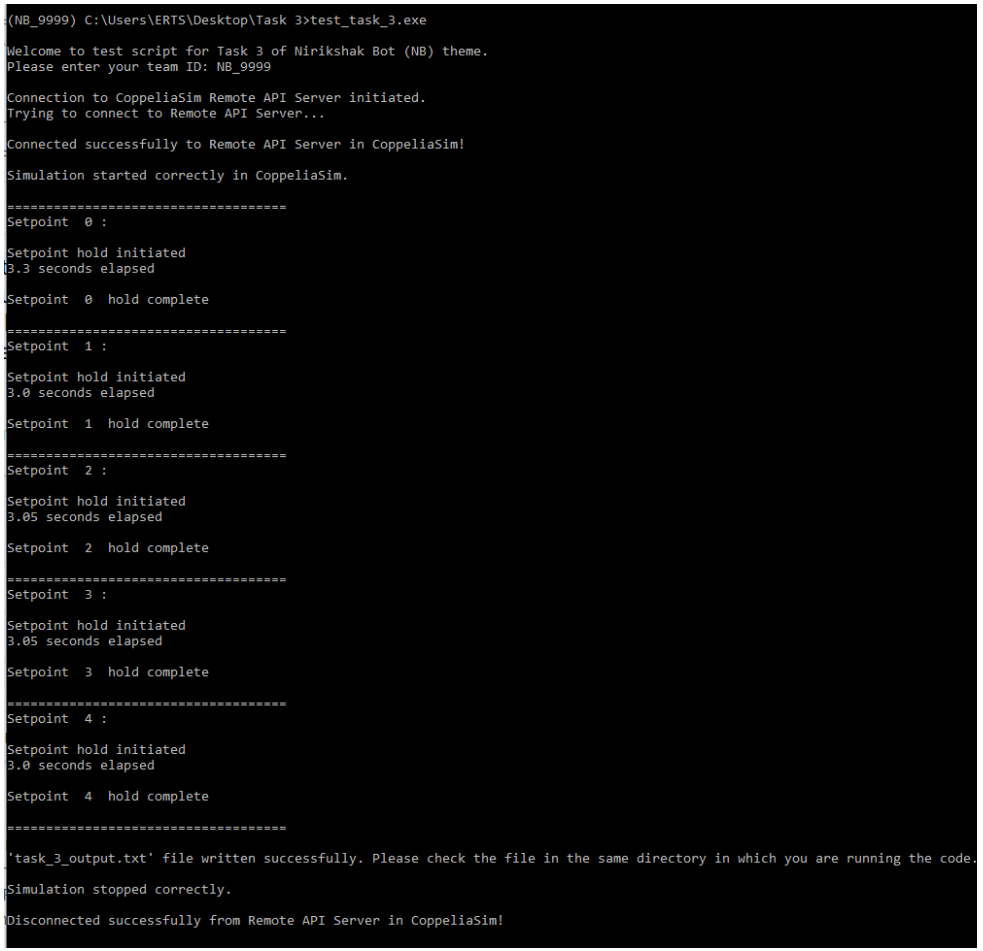


Figure 8: Output of **test_task_3.exe**.

- During **execution** of the file, you will find the output as shown in Figure 9.

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

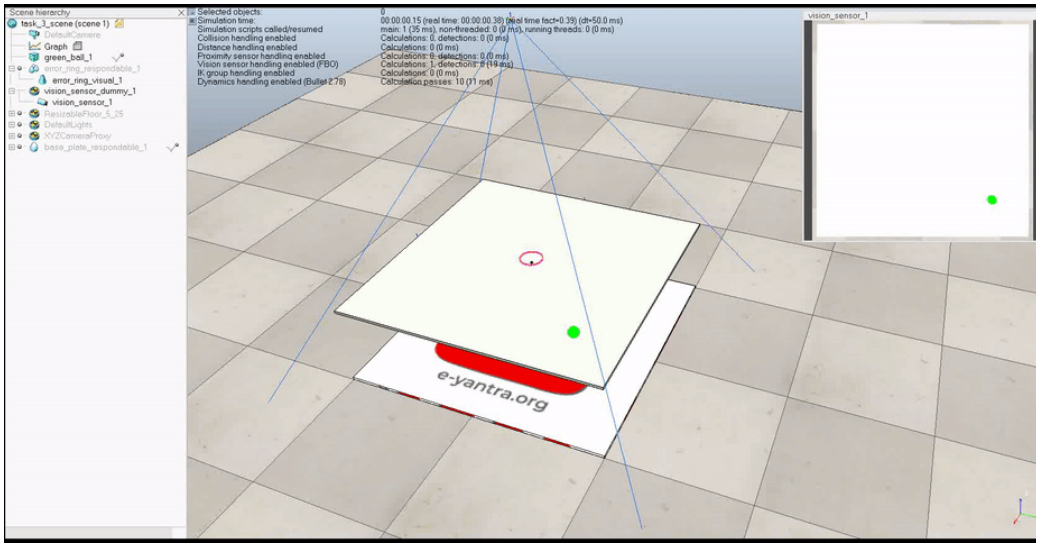


Figure 9: Execution of **test_task_3**.

- After execution of the file, `task_3_output.txt` will be **created in the same directory** in whic the `test_task_3.exe` was running.

NOTE:

- During the time when `test_task_3.exe` or `test_task_3` is running, your CoppeliaSim's simulation may lag.
- In this task, you will have to **optimize your code as much as possible to reduce this lag**.
- Even a **small lag** may **reduce the performance** of your control logic.
- Try to maintain it **as low as possible**.

Submission Instructions

For **Task 3 submission** you have to upload a **.zip** file. To create the appropriate file please follow instructions given below:

1. Create a new folder named `NB_<Team-ID>_Task_3` .

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

- For example: if your team ID is **9999** then you need to create a folder named `NB_9999_Task_3` .

2. Now copy and paste following files into this folder:

   - `task_3_scene.ttt` (with the **modified model** of Task 1C)
   - `task_3_output.xml` (Refer Figure 8 of Task 1C - Design Ball Balance Platform document to learn how to export CoppeliaSim scene as an XML file)
   - `task_3_output.txt` (generated after running `test_task_3.exe` OR `test_task_3`)
   - `task_3_simulation.avi` (generated by recording video of simulation from CoppeliaSim)
   - `task_3.py`
   - `task_2a.py`
   - `task_1b.py`
   - `task_1a_part1.py`

3. **Compress** this folder into **a zip file** and name it as `NB_9999_Task_3.zip` .

4. Now go to the eYRC Portal and follow the instructions to upload this **.zip** file for **Task_3** as shown in Figure 10.

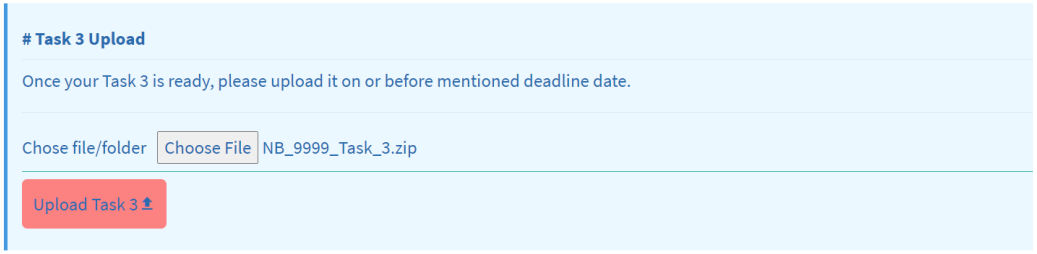
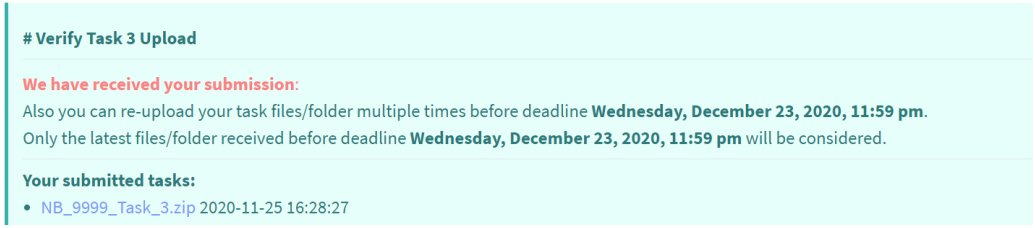


Figure 10: Submission of **NB_9999_Task_3.zip** file on eYRC portal.

NOTE: File names mentioned are case sensitive. Verify all the file names before creating the zip file.

---

5. After you have **successfully submitted Task 3 files**, you can verify the zip file uploaded from the **'Verify Task 3 Upload'** section on the eYRC portal. The same is shown is Figure 11.



Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Tips and Tricks to improve your Ball Balancing Platform Design

Introduction to Control Systems

Understanding Proportional Integral Derivative (PID) Controller

Balance ball on platform

Task 4

Task 5

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog

Figure 11: Successful submission of **Task 3** on eYRC portal.

ALL THE BEST!!