Task 5

Progress Task

[*Last Updated on:* ***19th January 2021, 15:00 Hrs***]

NOTE:

Before proceeding further, make sure you have gone through:

- Resources provided for this task and
- Rulebook
- Tutorial for implementing Perspective Transform with Open Maze

Teams should write the solution for this task so that they can easily modify it and solve Task 6 i.e. THE FINAL THEME IMPLEMENTATION.

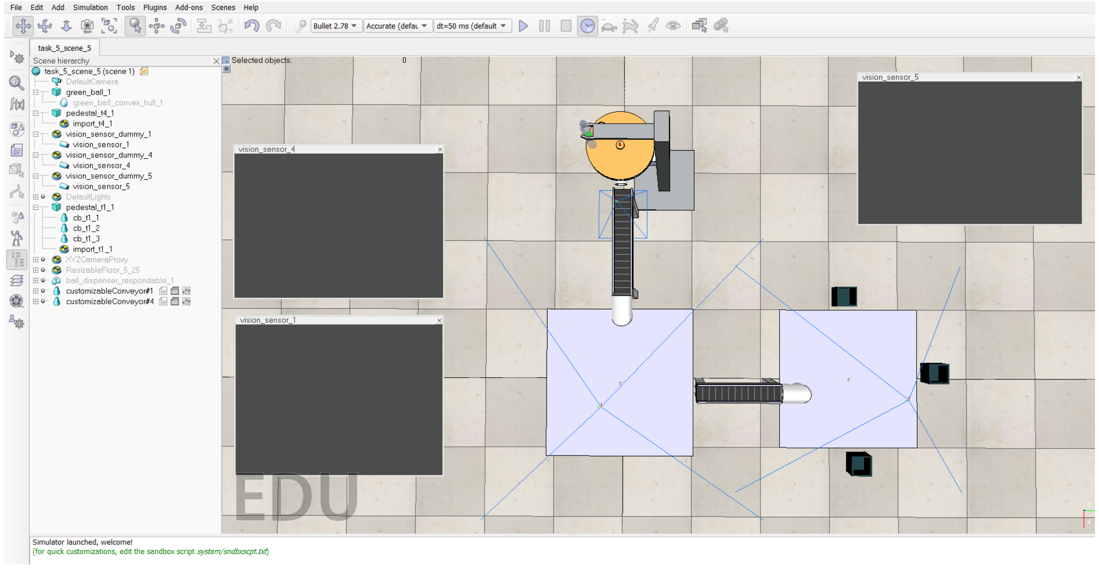
1. Problem Statement

Develop an algorithm **to navigate the ball through the maze** on top of **2 ball balancing platforms** and **deposit it in the collection box** of the given CoppeliaSim scene.

2. Given

1. CoppeliaSim's Scene File (`task_5_scene.ttt`)

- A scene file i.e. `task_5_scene.ttt` of CoppeliaSim software as shown in Figure 1.



```
Simulator launched, welcome!
(for quick customizations, edit the sandbox script system/sndbxscpt.txt)
```

Figure 1: CoppeliaSim scene file for Task 5.

- Objects along with their names and uses are provided in Table 1.

Objects	Name in the scene	Use(s)
Green Ball	*green_ball_1*	**Green color ball** that is supposed to be steered from one cell of the maze to the other.
Green Ball Convex Hull	*green_ball_convex_hull_1*	Used for **detecting collision** with the wall.
Vision Sensor Dummy	*vision_sensor_dummy_1,* *vision_sensor_dummy_4* *vision_sensor_dummy_5*	**Dummy** used to position and orient Vision Sensor.
Vision Sensor	*vision_sensor_1,* *vision_sensor_4,* *vision_sensor_5*	Works as the **camera** in the scene. The **output of vision sensor** is supposed to be used for **Image Processing**.
Pedestal	*pedestal_t4_1,* *pedestal_t1_1*	Used to provide the **necessary height** to the ball balancing platform.

Objects	Name in the scene	Use(s)
Collection Box	*cb_t1_1*, *cb_t1_2* *cb_t1_3*	Used to **collect the balls** from the **exit points** of **Table-1**
Import Dummy	*import_t1_1*, *import_t4_1*	Used to **position the Ball Balancing Platform** made by teams in Task 4B.
Ball Dispenser	*ball_dispenser_respondable_1*	Used to **dispense balls at regular intervals**.

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

- Floating view(s) of the output of ***vision_sensor_1***, ***vision_sensor_4*** and ***vision_sensor_5*** are also shown.
- Important Points to **NOTE**:
  - You are **NOT** allowed to **remove** the above mentioned objects.
  - You are only **ALLOWED to change the 'height/z-coordinate'** of the following objects:
    1. Customizable Conveyor
    2. Down Pipe
  - As mentioned in Task 4B, teams **CAN decide the resolution of Vision Sensors**.
  - Any other **change in the properties/position/orientation** of the above mentioned objects will lead to **POOR/NULL EVALUATION**.
- Teams will have to **import their ball balancing platform** from Task 4B.

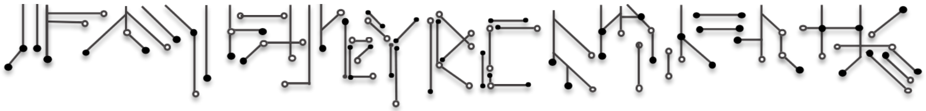
2. Main Script (`task_5.py`)

- The python script i.e. `task_5.py` **NEEDS** to be edited by the teams.

3. Ball Details (`ball_details.json`)

- This JSON file consists of the details of all balls that will be dispensed by the **Ball Dispenser (BD)**.

4. Maze Images for Table 4 and 1 (`maze_t4.jpg`, `maze_t1.jpg`)



eYRC 2020-21: Nirikshak Bot (NB)

- The team **should run this executable ONLY after they have completed** writing the `task_5.py` script.

3. Getting Started

- Download the following files in a directory to get started with the task. **Right-click** on the hyperlink and select **Save Link As...** option to download.
    - **task_5_scene.ttt**
    - **task_5.py**
    - **ball_details.json**
    - **maze_t4.jpg**
    - **maze_t1.jpg**
    - **task5_scene_details**

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

- Make sure you add following files in the same directory:
    - `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.
- Now, read the following instructions carefully.

4. Understanding the Task

A. CoppeliaSim Scene (`task_5_scene.ttt`)

(i) General Scene Properties:

- Export the model made by your team in Task 4B as a **CoppeliaSim Model File**. You can refer Figure 10 of Task 1C- Design Ball Balance Platform documentation in order to do so.
- 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.
- After you open `task_5_scene.ttt`, you have to **add this model file**.
- Use ***import_t1_1*** and ***import_t4_1*** dummies to position your ball balancing platforms. Refer Figure 2.

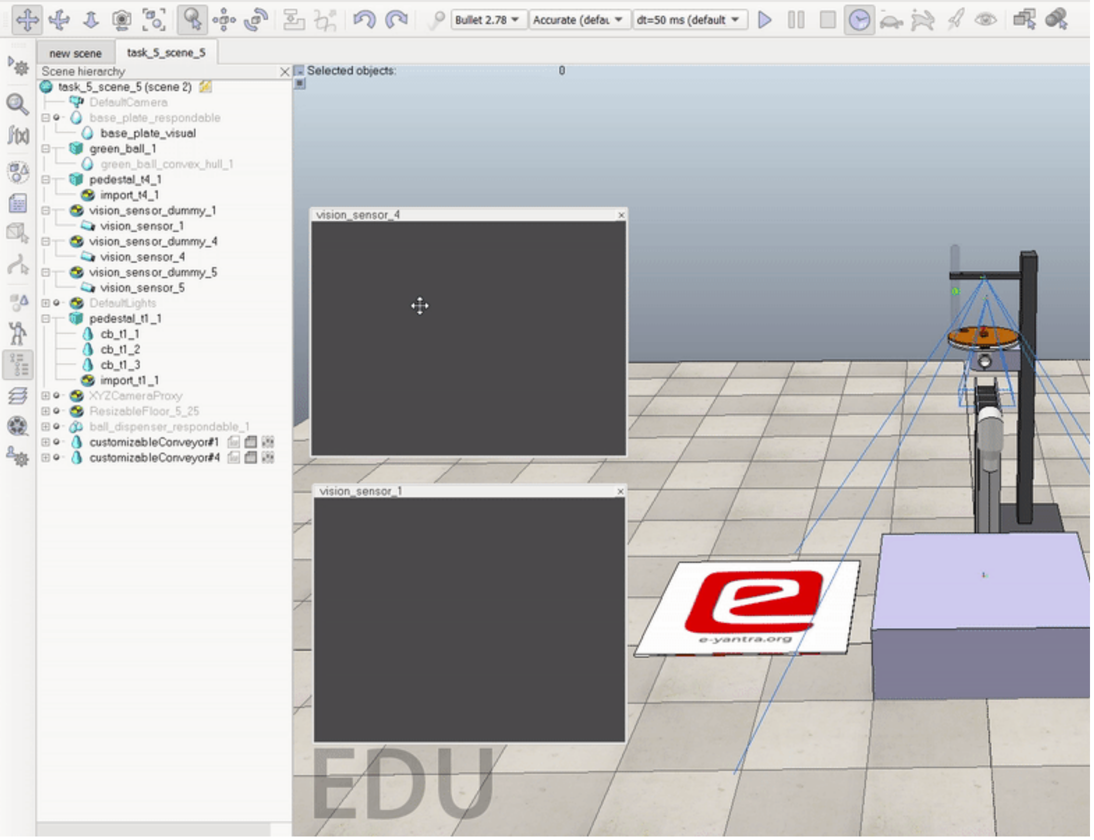


Figure 2: Using **import_t1_1** and **import_t4_1** dummies to position your ball balancing platforms.

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Task 4

Task 5

Tutorial on Perspective Transform with Open Maze

Progress Task

Scene Details

App for Grading of Progress Task

Submission Instructions

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:

- After positioning your model, make sure to **assign the entire model as a child of respective pedestal**.
- For example:
  - After positioning base_plate_respondable_t4_1, you will have to assign it as a child of pedestal_t4_1.

- From this task onwards, **you will have to follow the naming convention as mentioned in Table 2**. Make sure to refer the Theme Description to understand the table naming conventio

Object Type	No repeated occurrences	Repeated occurrences
respondables	**Insert '*_t<table_number>_1*'**. e.g. *base_plate_respondable_t4_1*, *base_plate_respondable_t3_1*	**Insert '*_t<table_number>_<quantity>*'** after the name. e.g. *servo_fin_respondable_t4_1*, *servo_fin_respondable_t4_2* etc.
visuals	**Insert '*_t<table_number>_1*'**. e.g. *yoke_visual_t4_1*	**Insert '*_t<table_number>_<quantity>*'** after the name. e.g. *yoke_visual_t4_1*, *yoke_visual_t4_2* etc.
force sensors	**Insert '*_<1st Initial of parent and 1st Initial of child>_t<table_number>_1*'**. e.g. *force_sensor_by_t2_1*	**Insert '*_<1st Initial of parent and 1st Initial of child>_t<table_number>_<quantity>*'** after the name. e.g. *force_sensor_by_t2_1*, *force_sensor_by_t2_2* etc.
joints (revolute, spherical etc.)	**Insert '*_<1st Initial of parent and 1st Initial of child>_t<table_number>_1*'**. e.g. *revolute_joint_ss_t1_1*	**Insert '*_<1st Initial of parent and 1st Initial of child>_t<table_number>_<quantity>*'** after the name. e.g. *revolute_joint_ss_t1_1*, *revolute_joint_ss_t1_2* etc.
dummies	**Insert '*_t<table_number>_1*'**. e.g. *dummy_t3_1*	**Insert '*_t<table_number>_<quantity>*'** after the name. e.g. *dummy_t3_1*, *dummy_t3_2* etc.

Table 2: Naming convention for respondables, visuals, force sensors, joints and dummies.

- Teams are **ONLY ALLOWED to change the 'height/z-coordinate'** of ***customizable conveyor*** and ***down_pipe***.
- Refer the Figure 3 to learn how to modify the height of conveyor belt.

Welcome to NB theme!

Rulebook ›

Task 0 ›

Task 1 ›

Task 2 ›

Task 3 ›

Task 4 ›

Task 5 ⌄

Tutorial on Perspective Transform with Open Maze

Progress Task

Scene Details

App for Grading of Progress Task

Submission Instructions

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

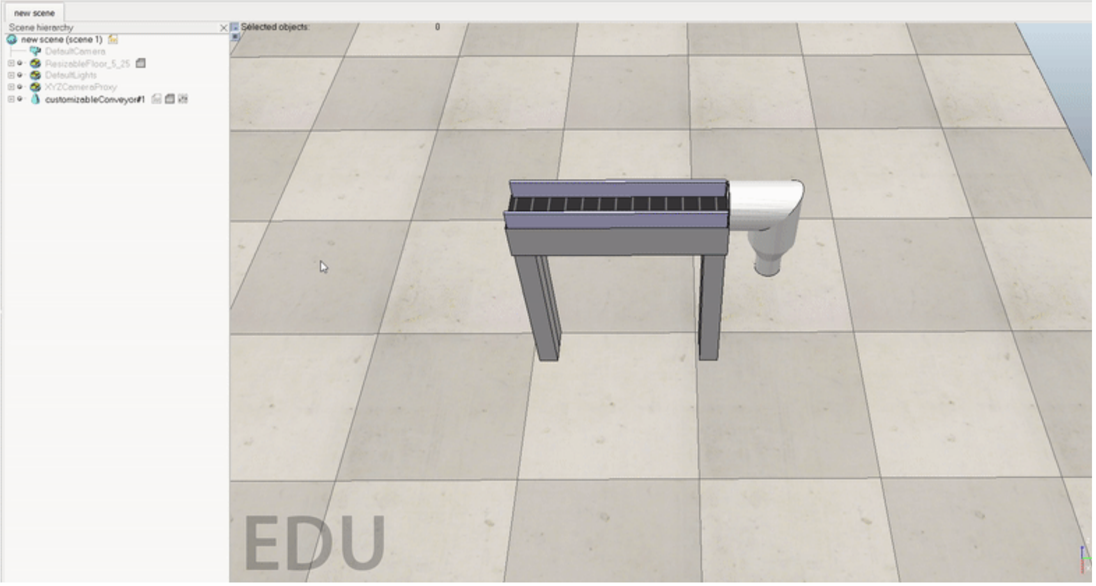


Figure 3: Modifying the height of conveyor belt.

NOTE:

- **NEVER** click on cross (X) icon next to customizable conveyor. Doing so **will disable the dialog box in the current scene**.
- The conveyor will **have to be inserted again** to bring the dialog box back.
- Teams are **NOT allowed** to modify any other property of the conveyor **such as length, width or speed**.

(ii) Collection and Collision Objects:

- For this task, you will have to **create collection and collision object** for **EACH** platform in the scene. Refer Table 3.

Sr. No.	Name of the Collection	Elements to be added in the collection
1	colliding_objects_t1	pegs_t1_1, walls_t1_1
2	colliding_objects_t4	pegs_t4_1, walls_t4_1

Table 3: Naming convention for **Collections**.

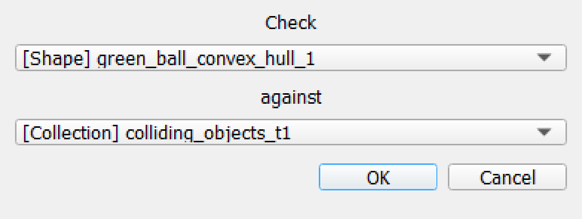
Welcome to NB theme!
Rulebook ›
Task 0 ›
Task 1 ›
Task 2 ›
Task 3 ›
Task 4 ›
Task 5 ⌄
Tutorial on Perspective Transform with Open Maze
Progress Task
Scene Details
App for Grading of Progress Task
Submission Instructions
Practice Task
Instructions for Task 6
Task 6 Scene Details
Coding Standard
Git and 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

- Click on **Calculation Module Properties** ƒ(x) and select **Add new collision object**. Refer Table 4 and Figure 4.

Sr. No.	Name of the Collision Object	Element to be checked	Element to be checked against
1	ball_collision_t1	green_ball_convex_hull_1	colliding_objects_t1
2	ball_collision_t4	green_ball_convex_hull_1	colliding_objects_t4

Table 4: Naming convention for **Collision Objects**.

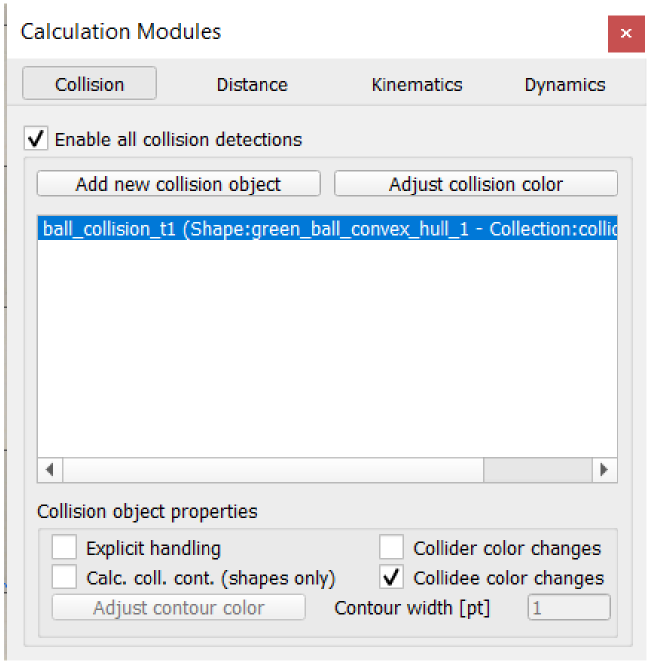
Collision Object, Selection of Entities



Check: [Shape] green_ball_convex_hull_1
against: [Collection] colliding_objects_t1

Figure 4: **Colliding objects** to be added in the Calculation Module dialog box.

- Make sure to check ONLY **Collidee color changes**. On addition of **ball_collision_t1**, the Calculation Module should be as shown in Figure 5.



Calculation Modules

☑ Enable all collision detections

ball_collision_t1 (Shape:green_ball_convex_hull_1 - Collection:collic

Collision object properties

☐ Explicit handling ☐ Collider color changes
☐ Calc. coll. cont. (shapes only) ☑ Collidee color changes
Contour width [pt] 1

Figure 5: Addition of ***ball_collision_t1*** in Calculation Module dialog box.

- Now click on **Adjust collision color -> Ambient/diffuse component** and set the **RED** and **GREEN** values as 1 whereas the **BLUE** value as 0.
- This will make sure that the color of collidee i.e. walls and pegs changes to **YELLOW** in case of collision.
- Repeat the same procedure for ***ball_collision_t4***.

(iii) Respondable Objects:

- The following objects (**apart from the ones given in the scene**) are **NECESSARY** to be kept a ***respondable*** for **ALL the tables**.

Sr. No.	Object Name	Global Mask	Local Mask
1	base_plate_respondable	✓	✓
2	servo_holder_respondable	✓	✓
3	yoke_respondable (ONLY the one which is connected to base plate)	✓	✓
4	top_plate_respondable	✓	✓
5	pegs	✓	**X**
6	walls	✓	**X**

Table 5: Necessary **respondable** elements.

(iv) Collidable Objects:

- The following objects (**apart from the ones given in the scene**) are **NECESSARY** to be kept a ***collidable*** for **ALL the tables**.

Sr. No.	Object Name
1	walls
2	pegs

Table 6: Necessary **collidable** elements.

(v) Renderable Objects:

- The following objects (**apart from the ones given in the scene**) are **NECESSARY** to be kept a ***renderable*** for **ALL the tables**.

Sr. No.	Object Name
1	top_plate_respondable
2	pegs
3	walls

Table 7: Necessary **renderable** elements.

NOTE:

- In order to set the **walls** as **renderable** and **collidable**, teams will have to using the following API in `createWall()` function of the Lua script:

```
sim.setObjectSpecialProperty(wallObjectHandle,
sim.boolOr32(sim.objectspecialproperty_renderable,
sim.objectspecialproperty_collidable))
```

- You will **not be able** to set an object's special property by repeating the API `sim.setObjectSpecialProperty()` twice.
- Hence it is necessary to use `sim.boolOr32()` to enable both the properties of the walls.

B. Python Script (`task_5.py`)

- The boiler plate code provided contains only 2 functions as follows:

1. `main(rec_client_id)`

Parameters	Details

Parameters	Details
Purpose	Teams are **free to design their code** in this task. The test executable will only call this function of task_5.py. `init_remote_api_server()` and `exit_remote_api_server()` functions are already defined in the executable and hence should **NOT** be called by the teams. The **obtained client_id** is passed from the executable to this function so that teams can use it in their code. If the executable is not running, teams should call the `init_remote_api_server()` in the default main function of the `task_5.py` .
Input Arguments	`rec_client_id` : [**integer**] ***client_id*** generated from start connection remote API (called by the executable).
Return	None
Example Call	`main(rec_client_id)`

Table 8: ***main()*** function to be edited by teams in **task_5.py**.

2. `send_color_and_collection_box_identified(ball_color, collection_box_name)`

Parameters	Details
Purpose	1. This function should only be called when the task is being evaluated using test executable. 2. The format to send the data to the evaluation script is as '**color::collection_box_name**'
Input Arguments	`ball_color` : [**string**] Color of the ball identified from vision_sensor_5. `collection_box_name` : [**string**]:
Return	None
Example Call	`send_color_and_collection_box_identified(ball_color, collection_box_name)`

Table 9: ***send_color_and_collection_box_identified*** function to **NOT** be edited by teams in **task_5.p**

NOTE: You should **not use** `sys.exit()` in **task_5.py** or any other dependent Python files, as it will stop the execution and will not generate any output files while testing your solution with **test_task5_exe** file.

---

5. Expected Output

- Teams are provided with a **maximum time limit** of **180 Simulation Seconds** for this task.

  NOTE:

  - Remember, the task is to ***navigate the ball from one table to another by balancing the ball on the platform tables***. So, make sure your solution doesn't let the ball slide along or touch each wall of the maze without caring about the collisions that will result.
  - For the team's submission to be considered VALID, the **average Real Time Factor (RTF) of the entire run** of task 5 should be **>=0.5 (greater than or equal to 0.5)**.
  - Since the entire theme implementation, as per Rulebook has been allotted a maximum of **480 simulation seconds**, this decision has been taken for the benefit of students.
  - **Having a low RTF** will **prevent the teams from testing their solution again and again** since the real time would be huge.
  - For example:
    - If a team's solutions has a **RTF of 0.1** and the **entire simulation is taking 400 simulation seconds**, in **real time it will take 4000 (FOUR THOUSAND) seconds** which is around **66.67 minutes** per run.

- On running `task_5.py` , following should happen:

  1. **Maze array** of both the tables should be **transferred to the lua's customization scripts**.
  2. **Maze should be generated** on top of **both the tables** in the CoppeliaSim scene.
  3. **Color of the ball** should be **determined from *vision_sensor_5*,** as soon as the ball is detected.
  4. Color of the ball should now be compared with the contents of the ***ball_details.json*** to **find the correct Collection Box**.
  5. **Path should be calculated** for both the mazes.
  6. Paths **should be transferred** and **drawn on top of the mazes**.
  7. Path on top of **Table 4 should be traversed**.
  8. Ball should be **passed to conveyor belt**.
  9. Path on top of **Table 1 should be traversed**.
  10. Ball should be **deposited in the designated collection box**.

- Refer Figure 6 to understand the flow mentioned above.

Welcome to NB theme!

Rulebook

Task 0

Task 1

Task 2

Task 3

Task 4

Task 5

Tutorial on Perspective Transform with Open Maze

Progress Task

Scene Details

App for Grading of Progress Task

Submission Instructions

Practice Task

Instructions for Task 6

Task 6 Scene Details

Coding Standard

Git and GitHub

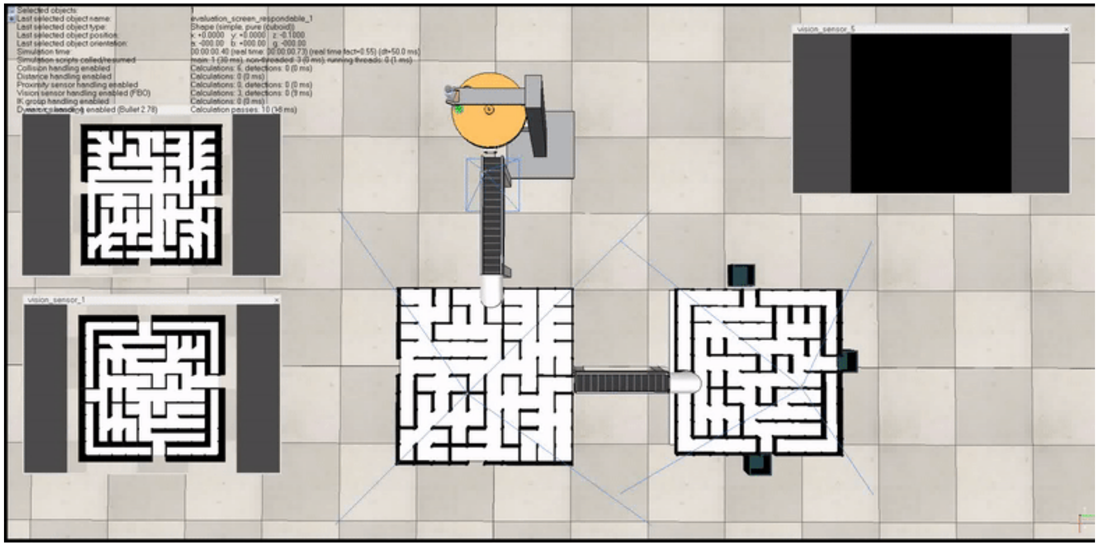
Live Session 1 - 24th October 2020

Live Session 2 - 21st November 2020

Live Session 3 - 12th December 2020

Live Session 4 - 10th January 2021

Changelog



```
Selected objects:                          1
Last selected object name:                 evaluation_screen_respondable_1
Last selected object type:                 Shape (simple, pure (cuboid))
Last selected object position:             x: +0.0000   y: +0.0000   z: -0.1000
Last selected object orientation:          a: -000.00   b: +000.00   g: -000.00
Simulation time:                           00:00:00.40 (real time: 00:00:00.73) (real time fact=0.55) (dt=50.0 ms)
Simulation scripts called/resumed          main: 1 (30 ms), non-threaded: 3 (0 ms), running threads: 0 (1 ms)
Collision handling enabled                 Calculations: 6, detections: 0 (0 ms)
Distance handling enabled                  Calculations: 0 (0 ms)
Proximity sensor handling enabled          Calculations: 0, detections: 0 (0 ms)
Vision sensor handling enabled (FBO)       Calculations: 3, detections: 0 (9 ms)
IK group handling enabled                  Calculations: 0 (0 ms)
Dynamics handling enabled (Bullet 2.78)    Calculation passes: 10 (16 ms)
```

Figure 6: Initial simulation flow of task 5.

---

6. Testing the Solution and Submission Instructions

- Refer the App for Grading of Progress Task document for testing your solution `task_5.py` file
- Refer the Submission Instructions document.

ALL THE BEST!!