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

Tutorial

Implementing Perspective Transform with open mazes

[*Last Updated on:* ***15th January 2020, 22:00 Hrs***]

In this tutorial, we will discuss a methodology to apply a Perspective Transform on open maze images.

This methodology will be useful when you try to apply the Perspective Transform in Task 5 and the Final Theme Implementation.

1. Introduction

- So far in this theme, we have implemented the Perspective Transform on closed maze images (such as Figure 1).

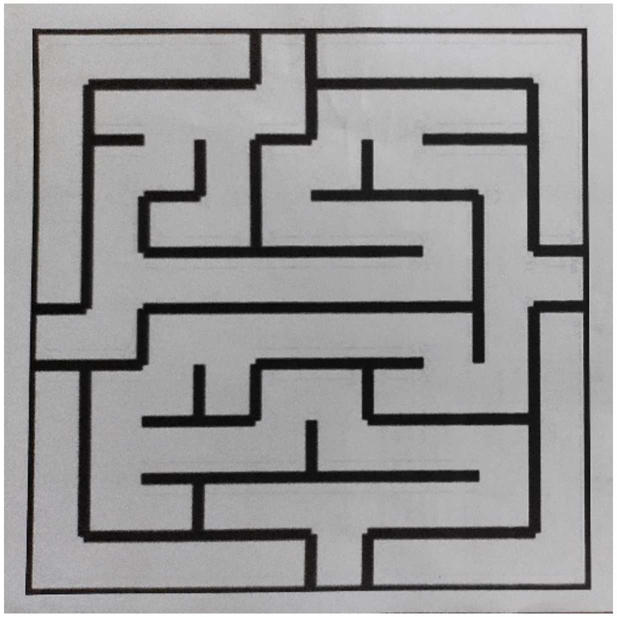

Figure 1: Maze

- You have also applied the Perspective Transform in **Task 3** and **Task 4B** where you will have used the `applyPerspectiveTransform()` to isolate the top plate of the ball balancing platforr in order to track the position of ball.
- In both these cases, we were dealing with closed contours (shown in Figure 2).

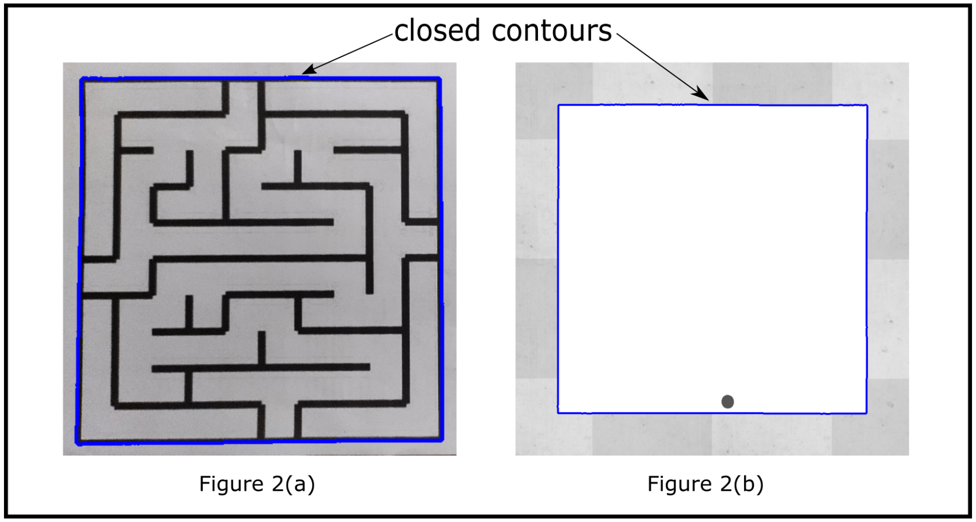

Figure 2(a)

Figure 2(b)

Figure 2: Closed Contours

- However, in **Task 5**, you have to work with rendered mazes. Hence the images you obtain fror the vision sensor will resemble Figure 3.

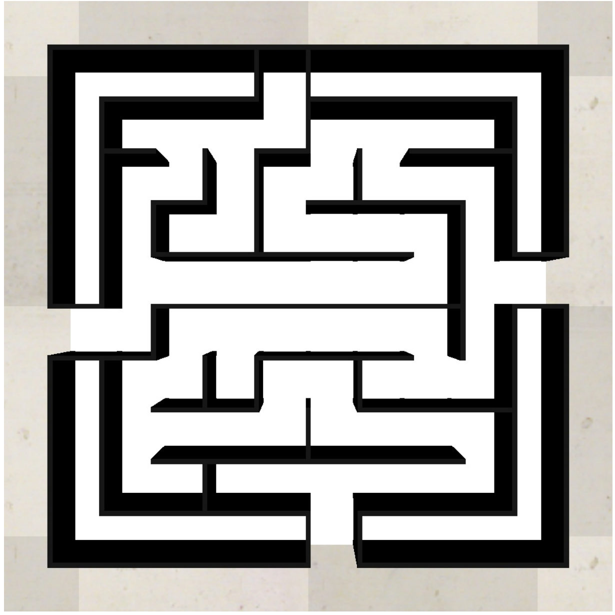

Figure 3: Maze with open exits

- In this case, since you have carved the exit points in the maze, hence it will not be possible to obtain a closed contour as depicted earlier. Hence the approach for **Perspective Transform** needs to change in this task.

2. Methodology

- We will discuss the new approach in this section.
- In order to apply a **Perspective Transform** to isolate only the maze in the image, we need to find the four corner points of the maze (depicted as blue dot in Figure 4).

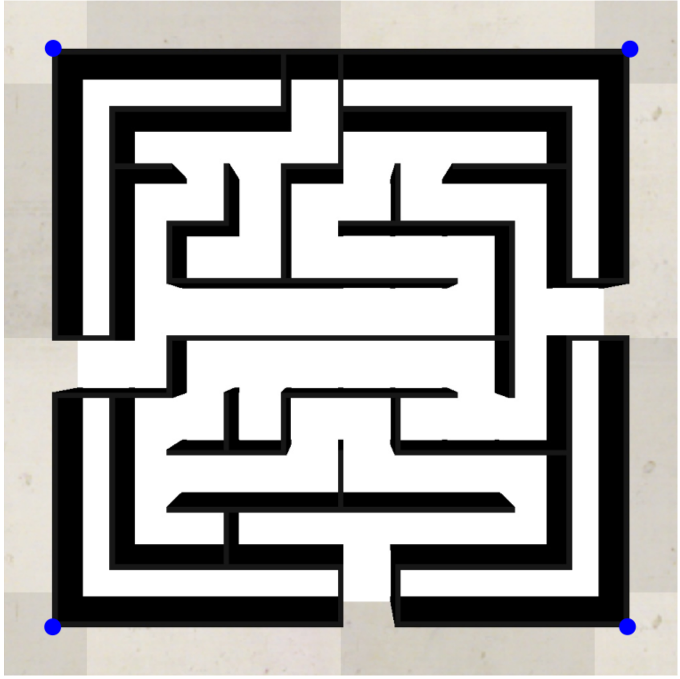

Figure 4: Corner points of maze

- As we discussed earlier, in prior cases since we were dealing with a closed maze, hence these corner points could be easily obtained by first obtaining the outer closed contour of the maze
- In this case, if we try to obtain the contours for the maze, we will obtain contours as depicted Figure 5

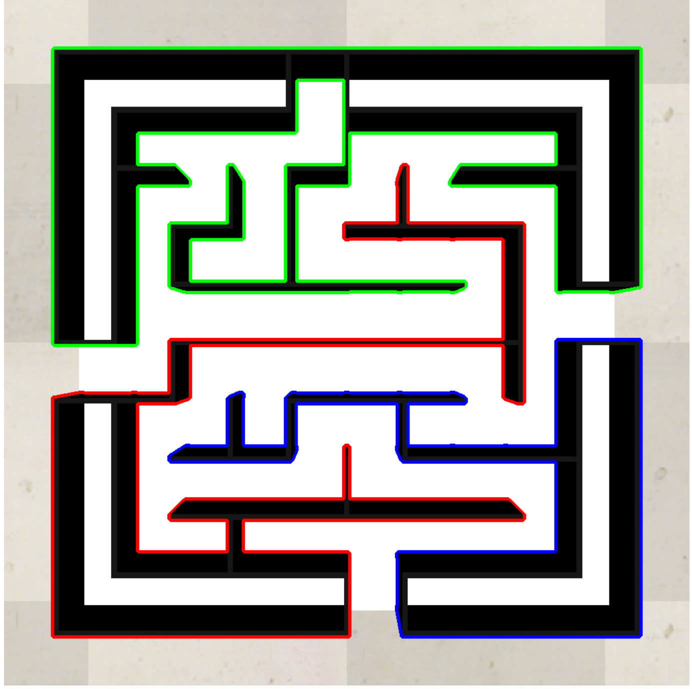

Figure 5: Contours in open maze

- In order to find the corners of the maze, you will first need to find the bounding boxes of all th highlighted contours (shown in Figure 6).

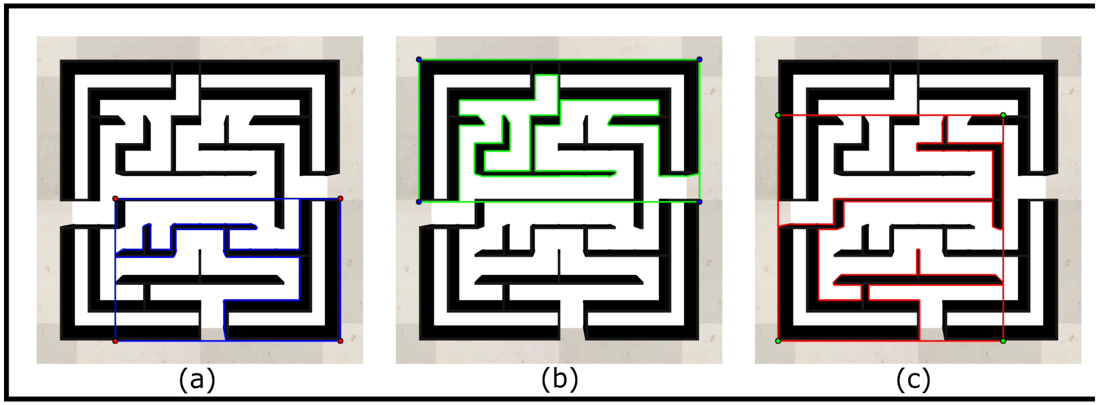

Figure 6: Bounding Boxes for all the contours

- In Figure 6, you will notice that we have obtained the bounding boxes of all the contours in th given maze. The corners of all 3 bounding boxes have been highlighted in the Figure 6.
- There are a total of 12 corner points highlighted in Figure 6 (4 for each bounding box). You wil notice that out of these 12 corner points, 4 of them coincide with the corner points of the maz

itself. Hence if these 4 corner points are chosen while applying the perspective transform, you will be able to isolate the maze in the image.

- Hence you have to write an algorithm to collect all these corner points in a python list and the select 4 of the corner points from that list that coincide with the corner points of the maze.

3. Conclusion

- Hence we have demonstrated the methodology to use for implementing perspective transfor for open mazes. You have to work on this implementation as a part of Task 5.
- The final output of the perspective transform should be as follows (shown in Figure 7)

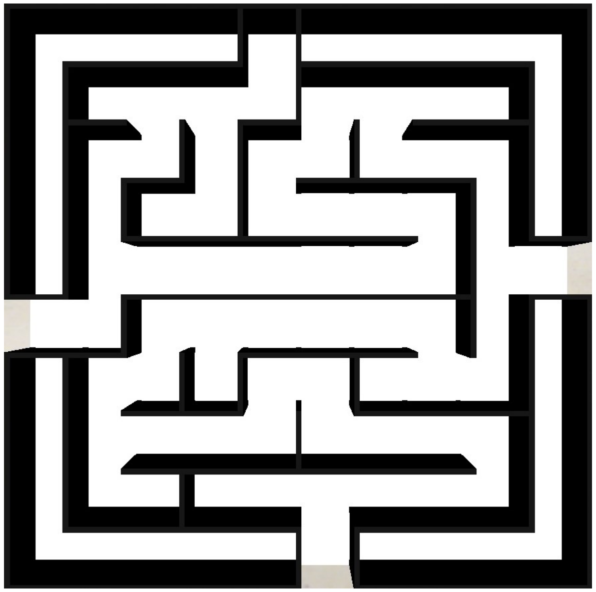

Figure 7: Final Output

ALL THE BEST !!