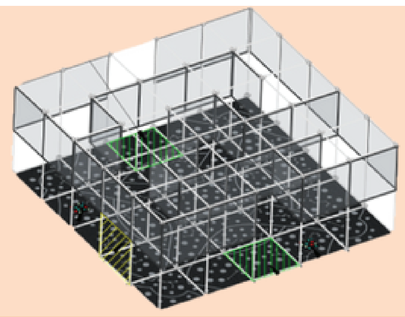
DRONE NAVIGATION AND MAZE MAPPING

[-CAMP CHALLENGE-]



Part 1: Using Python, program a TelloEDU drone to **{explore the maze}** and **{create a map}** (nodes+edges and mission pad locations) of an unknown 5x10 maze, and **{exit the maze}** to gain extra points. The team is also given points for correct nodes on the map.
Part 2: Program **{a swarm of 4 drones}** to land on mission pads on the maze floor (known locations). Additionally, use **{computer vision}** to **{recognise}** a medic box on the maze walls for extra points.

[-POSSIBILITIES-]

There are various maze-solving and maze-exploring algorithms. For example, the most popular ones we found in our pre-competition research were **Depth First Search (DFS)**, **Breadth First Search (BFS)**, and **Flood Fill**, amongst others. We can also use Tremaux's algorithm to determine which parts of the maze we have navigated through and decide which path to go for next. (note: Tremaux's algorithm is similar to DFS, the difference being the method of marking explored/unexplored paths)

For the second part of the challenge, given the small size of the map, manually programming individual drone paths was the best option as it was fast and simple. As part of our strategy, we also recentered the drones at critical turning points to avoid crashing due to its instability.

[-DETECTION-]

- Ensure that the machine is trained on images of **different scales, orientations and backgrounds of the medical kit (wall colour)** to increase the accuracy of detection
- Maintain a balance between having **enough data and avoiding** excessive training which would result in **overfitting**
- Maximized the **number of epochs** until the machine is too familiar with only the images it was trained on, which can improve accuracy

[-OUR STRATEGY-]

We chose **DFS** as our strategy for solving the maze because it is faster than BFS and other methods. The maze we were dealing with was highly connected and had potential loops, which would be easier to navigate using DFS than BFS.

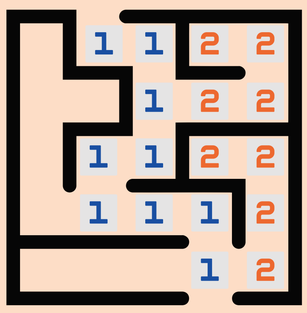


Figure 1: illustration of how the clear and obstructed paths were graphed

Our DFS algorithm analysed each node that was possible to move to, always going to the leftmost unexplored direction and **counting it as 1**. Whenever we reached a dead end, we instructed the robot to **mark everything in the current path to the nearest turning node as 2**. Taking inspiration from Tremaux's Algorithm, the turns and their locations were stored. This prevented the robot from making too many unnecessary turns, and instead remember which route we had already explored, allowing us to retreat to an unexplored branch.

During the second phase, we deployed the drones at intervals to ensure that they maintained a safe distance from each other and avoided collisions. Additionally, we programmed the drones at the rear with completed objectives to conduct the next drone's mission as a precautionary measure. This approach enhanced the reliability of our operation, as we acknowledge that drones may encounter challenges while navigating the maze.

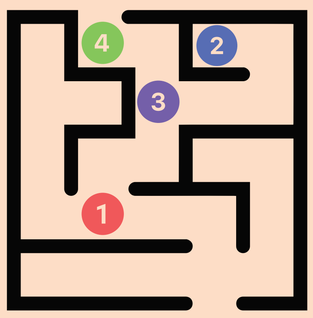
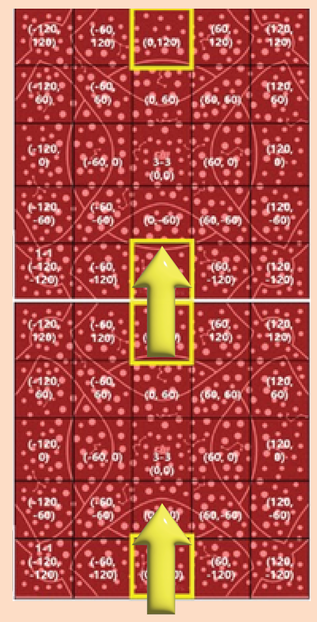


Figure 2: illustration of 4 drones at an instance in time, when drone deployment is done in intervals

[-DIFFICULTIES-]

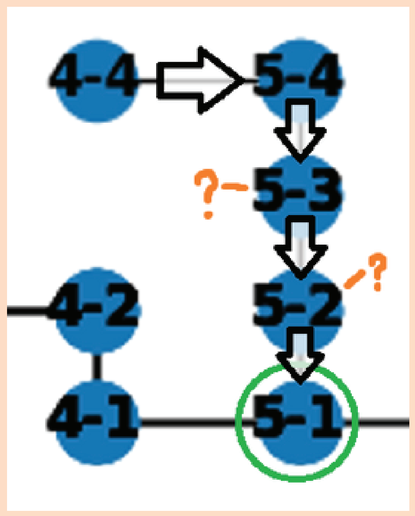


1 ***[Moving from 1st map to 2nd map]***

The drone reads patterns on the DJI Tello mat to determine its coordinates. However, the maze has two mats, hence two origins, and a jump in coordinates when moving from one maze to another.

Our solution:

- *We mapped the bottom half of the maze first before mapping the top half*
- *We hard-coded the drone to move forward and reset the coordinates afterwards*



2 ***[Dead ends]***

- The program could not identify which node to continue from after returning from a dead end.
- Time is wasted if the drone has to turn back and start identifying new available paths

Our solution:

- *Store the location of the previous empty junction and mark the path the drone just travelled as a 'used' pathway.*
- *The drone would go down the 'unused' pathway after that*

[-CONCLUSION-]

- Our progress was limited by the extent of experimentation and testing
- Allowing each member to work on different parts and cross-checking when necessary allowed for a diversity of ideas

Members:
Benita Yalini Benedict, Raffles Institution
Eric Cai Zhitong, Raffles Institution
Lim Wei En, Raffles Institution
Neo Jin Kai Jonathan, Raffles Institution
Wayne William Loo Shyan, Raffles Institution
Yuan Hao Zhe, Raffles Institution

YDSP Young Defence Scientists Programme
DSTA Defence Science & Technology Agency
DSO NATIONAL LABORATORIES