e-Yantra Robotics Competition - 2018
Theme and Implementation Analysis – Ant Bot

426

Team leader name	Rohan Katkam
College	National Institute of Technology Karnataka, Surathkal
Email	roh10162@gmail.com
Date	02/01/2019

Scope and Preparing the Arena

Q1. a. State the scope of the theme assigned to you.

(5)

The theme assigned to our team is Ant Bot. The Ant Bot is to collect supply blocks (Honey Dew, Leaves, Wood) from the Shrubs Area (SA) and place them in the AntHills (AH) as required. It also removes Trash from the AH. The Ant Bot is to give preferential treatment (highest priority) to the Queen AH (QAH). SIMs are ArUco IDs which give information about the service required.

One of the prominent applications of the Ant Bot is in the field of autonomous delivery. Such robots can be placed in the industrial manufacturing line to transport goods from one area in the factory to another. Using its image processing capabilities, it can be programmed to segregate items into categories and even detect faulty items.

With modifications, the robot can follow a route specified by GPS rather than a line which can allow for delivery of purchased items to consumers.

b. Upload the Final Arena Images.

(20)

Please check the folder *426_Task3* to view the final arena images

Building Modules

Q2. Identify the major components required for designing the robotic system for the theme assigned to you.

(5)

Mechanical Systems:

1. DC Motors
2. Chassis: body of the robot
3. Wheels: rear wheels and caster wheel
4. Arm Manipulator (Pick-and-place mechanism): consists of servo motors

Electronic Systems:

1. Raspberry Pi
2. Arduino Nano
3. Line-following sensor
4. PiCam

Power Management Systems:

1. L298N Motor Driver
2. Rechargeable 2200mAh LiPo battery
3. 10000mAh Power bank

Power Management

Q3. a. Explain the power management system required for a robot in general and for the theme assigned to you in particular.

(5)

Power needs to be supplied to the actuators, sensors, and the computing device integrating the actuators and the sensors.

The Raspberry Pi requires a power supply providing 2A at 5V.

The Arduino Nano can be powered by an unregulated power supply (6-20V) or a regulated power supply of 5V. On average, the Nano draws a current of 280mA.

These two computing devices can hence be powered by the Intex© power bank which has two ports capable of supplying 5V at a maximum current of 2A.

The 100rpm DC motors require 12V with a maximum current of 2,5A and a running current of 1.2A. The servo motors usually operate at 5V. The micro servo motor draws several hundred mA and the standard servo draws about 1A when moving (with load).

The LiPo battery has a C rating of 2 and a capacity of 2200mAh at an operating voltage of 11.1V. This means it can deliver a maximum current of 4.4A at a time which is sufficient to drive the DC motors. To power the servo motors, we can use the 5V@1A port from the Intex© power bank.

The sensors can be powered by the Raspberry Pi or the Arduino Nano accordingly.

b. Can there be a single power supply for your robot? - Yes/No/Don't know. Please elaborate/justify your answer choice.

(5)

No, there cannot be a single power supply for the robot. As mentioned above, the Raspberry Pi. Arduino Nano, and servo motors operate at 5V. However, the DC motors operate at 9-12V.

It might be possible to use a voltage regulator to step down the voltage from 12V to 5V. But if we were to use a single power supply, it would need to source a maximum of 6A at some point of time. The dimensions for such a power supply make it infeasible for use on the robot.

Design Analysis

Q4. Team have to design a robot which traverses the arena following a given path.

a. How will you design a robot to traverse the arena given in the rulebook?

(5)

The path-planning technique is as follows:
There are 15 nodes in the arena. We store each node as a vertex in a directed graph where the weights of each edge is the orientation required to move from one vertex to the next, i.e, **N**orth, **S**outh, **E**ast, and **W**est. Now to traverse from one node, say *src,* to another, say *dest*, we search for all possible paths from *src* to *dest* using backtracking. Note that there is only one possible path from *src* to *dest* because the graph created is a tree.

North is taken as the direction from the *START NODE* to the *CENTRAL NODE*. At the start of the run, the robot is placed at the *START NODE* and is facing the *CENTRAL NODE*. Hence at the start of the run, the robot is aligned in the north direction. The initial alignment of the robot is stored in a global variable and continuously updated as the robot turns left, right, or moves in the opposite direction. Likewise, the node in which the robot is placed at the start of the run, i.e *START NODE*, is stored in a global variable and is continuously updated as it moves to different nodes.

With these variables and the graph created, the robot can determine its present location, alignment, and the path to be taken to reach a particular node.

The line sensor can be used to detect the presence of a line or node as mentioned in the *Environment Sensing* section (Q.5a). Since it is an analog sensor that outputs values from 0.18V-2.2V, we can use it to accurately follow the line, making incremental corrections depending on the values read from each of the 3 channels line-sensors.

Inputs: Line sensor placed at the bottom of the chassis
Outputs: DC Motors placed on the sides of the chassis
Overall Algorithm:

1. Robot is placed at *START NODE.* Go to *CENTRAL NODE.*
2. Detect SIMs and identify ArUco IDs.
3. Determine QAH and service it first, lighting appropriate LEDs. Return to *CENTRAL NODE.*
4. Service remaining AHs, lighting appropriate LEDs. Return to *CENTRAL NODE.*
5. Go to *START NODE.* Sound the buzzer to indicate task completion.

b. How many actuators do you feel are sufficient for designing a pick and place mechanism? If you are going to use additional actuators (apart from those provided in the kit), how and for what purpose do you plan to use them?

(5)

For designing a simple pick-and-place mechanism, two actuators are required, both of which can be servo motors. One micro servo motor can be used to grasp the supply block and a standard servo can be used to lift the whole mechanism up.

Additional actuators add to the complexity of the pick-and-place mechanism but also increase its functionality. The mechanism previously discussed can pick only one block at a time. Another mechanism can have a DC motor that runs a conveyor belt. This can allow multiple blocks to be picked up without having to finish servicing the Anthill.

Environment Sensing

Q5.a. Explain how you will use the Line Sensor to decide the course of traversal (identifying line and nodes).

(5)

The line sensor given outputs a low voltage value (0.18V) when it is above a bright surface and a high voltage value (2.2V) when it is above a dark surface. The dimensions of a node is 3cm x 3cm and the width of the line is 1.2cm. Hence, the node's thickness is larger than the width of the line. This fact is used to determine if the robot has reached a node. When the line sensor is above a node, majority of the channels in the sensor will give a low value whereas when it is above a line, majority of the channels will give a higher value.

In addition to distinguishing between a node and a line, the orientation of the robot is important when travelling from one node to another. We have taken north to be the direction from the *START NODE* to the *CENTRAL NODE*. So for example if the robot has to move from the *START NODE* to the *CENTRAL NODE*, it needs to move towards the north.

There are 15 nodes in the arena. We store each node as a vertex in a directed graph where the weights of each edge is the orientation required to move from one vertex to the next, i.e, **N**orth, **S**outh, **E**ast, and **W**est. Now to traverse from one node, say *src,* to another, say *dest*, we search for all possible paths from *src* to *dest* using backtracking. Note that there is only one possible path from *src* to *dest* because the graph created is a tree.

b. Would the webcam be a better choice of camera over the PiCam? Explain.

(5)

The PiCam is run by the GPU of the Raspberry Pi whereas the USB webcam is run by the CPU. Hence running the PiCam does not load the CPU but running the webcam loads the CPU. Since it is directly interfaced to the Pi via the CSI, it can give a better framerate and resolution.

However, the PiCam can only be used by the Pi and not by other computers, unlike the webcam. The PiCam is also more delicate and more expensive than the webcam. The ribbon cable

connector for the PiCam is stiff so it prevents the PiCam from being positioned in a certain manner.

Ultimately, the choice of a PiCam or webcam depends on the requirement: if high resolution and framerate is required, the PiCam is a better choice. It also has in-built libraries which makes it easier to use than a webcam.

However, the webcam is sturdier and can be used when there is not a huge emphasis on image resolution and processing.

c. What other sensors will the robot require to complete its task successfully?

(5)

1. **Gyroscope sensor**: For angle, pick and place mechanism for arm manipulator.
2. **Accelerometer sensor**: For measuring the motion of the arm.
   Both the sensors mentioned above together can provide an adequate estimate of the arm's position and orientation.
3. **Proximity sensor**: for avoiding crashing into walls and incurring penalty.
4. **Distance sensor**: Like ultrasonic sensor can help calculate distance that is required in regions like AH. Since it is surrounded by walls, it can stop a required distance in order to assist pick and place mechanism.
5. **Overhead camera** (Image Sensor): Besides the PiCam, this will be mounted away from the robot on a stationary position at a height so it can map the entire arena, identify positions of nodes, identify and decode the SIMs (as well) and relay this information back to the bot.

d. Explain the strategy you will follow to detect and indicate the SIM placed around the Central Node (This includes traversing strategy to reach different SIMs).

(4)

1. As the task begins, the bot traverses from the start node to the central node and is left facing in the North direction in which it arrived at the central node. The PiCam is placed on a support that extends from the robot's body so that it has a better view of the SIMs.
2. The bot rotates by 45 degrees clockwise.
3. The Pi Cam always faces in the forward direction with respect to the bot. After Step 2, the Pi Cam is able to detect SIM 1. It will be detected with the help of corresponding functions from ArUco library. It will associate this with AH1, decode the information and store it.
4. Since the SIMs are placed symmetrically around the central node, the bot is rotated by 90 degrees hereafter to detect SIMs 2, 3 and 0.
5. The bot will associate the SIMs detected with the AHs 1, 2, 3 and 0 in that order.
6. The bot will finally orient itself to its initial direction (North) by rotating 45 degrees clockwise.
7. Upon decoding the information from all the SIMS, the bot will then begin its task to serve QAH followed by other AHs.

Testing your Understanding (Theme Analysis and Rulebook-related)

Q6. a. If at a given SIM location ArUco ID is found to be 76 (Decimal), what is the Ant Hill Number and type (Regular Ant Hill or Queen Ant Hill) and what are the Service Requirements of this Ant Hill?

(3)

Ant Hill Number: Regular Ant Hill
Ant Hill Type: Ant Hill 2 (AH2)
Service Requirements:
Serv 2 – Supply Required (Honey Dew - H)
Serv 1 – Supply Required (Leaves – L)

b. Is SIM0: 25, SIM1: 60, SIM2: 217, SIM3: 226, a possible combination of SIMs to be placed on the arena? If not explain with reasons.

(3)

SIM2 and SIM3 both have their QAH bits (the 7^{th} bit) set to 1. This means that both AH2 and AH3 are QAHs but according to the rulebook, there is a maximum of only one QAH. Consequently, SIM0: 25, SIM1: 60, SIM2: 217, SIM3: 226 is not a possible combination of SIMs to be placed on the arena.

c. What are the different conditions that indicate end of a run?

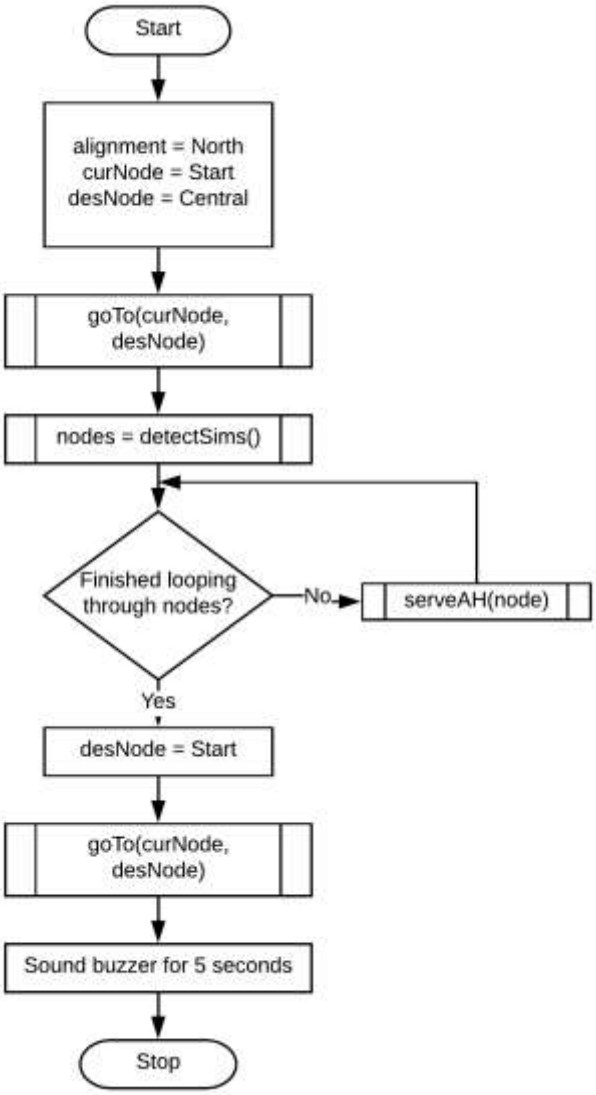
(3)

A run ends when any one of the following conditions are satisfied:

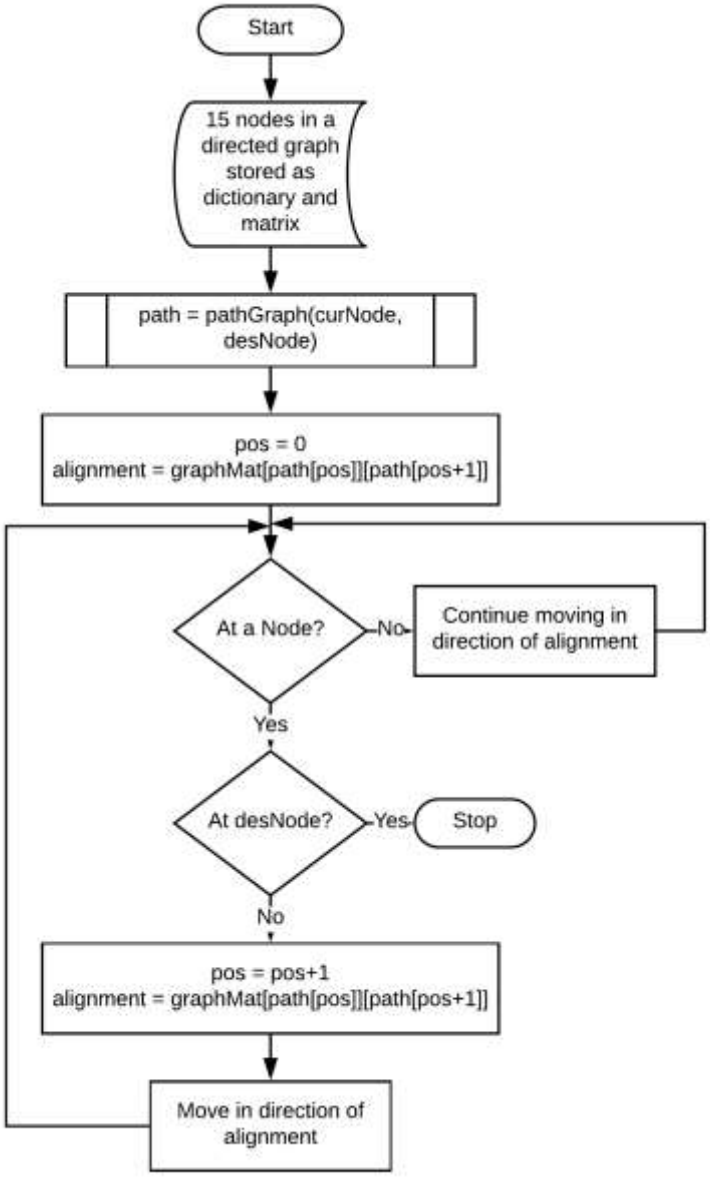
1. The robot completes the task and turns on the buzzer for 5 seconds after returning to the *START* position.
2. The robot exceeds the maximum time limit of 10 minutes (600 seconds) in completing the task.
3. The robot needs to be repositioned but the team has exceeded the maximum number of repositions allowed for each run (two repositions each run).

Algorithm Analysis

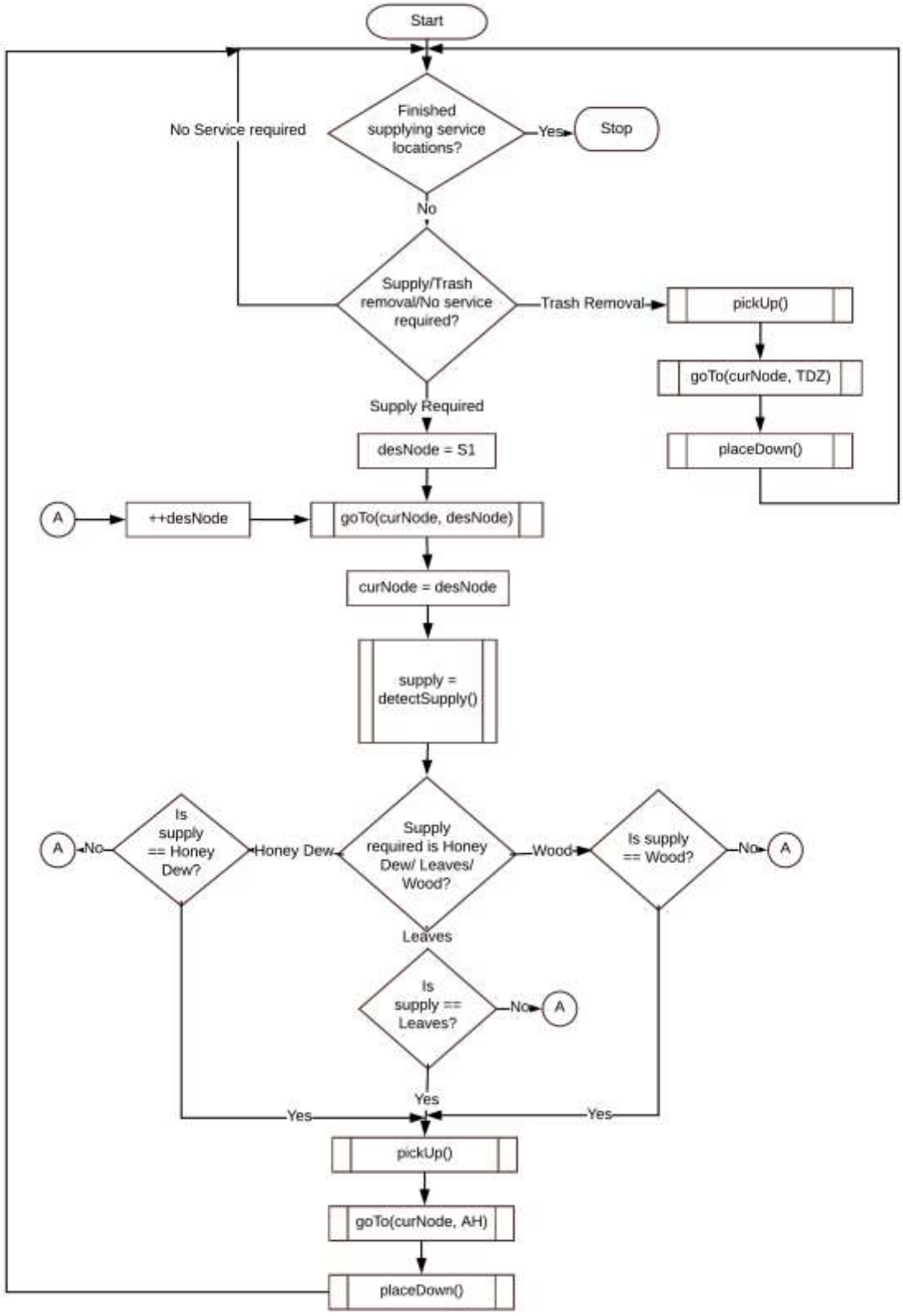
Q7. Draw a flowchart illustrating the algorithm you propose to use for theme implementation. (10)



Overall Flowchart



Flowchart for goTo() function



Flowchart for serveAH()

As indicated from the captions, the first flowchart shows a high-level glimpse of the program. The variable *alignment* indicates the alignment of the robot with respect to the North as discussed in previous answers. *curNode* is the node on which the bot is present or is moving from. *desNode* is the node to which the bot is moving.

goTo(curNode, desNode) is a function created to enable the robot to traverse from *curNode* to *desNode*. As mentioned in previous answers, all the nodes are considered vertices of a graph. Each node has a corresponding list of adjacent vertices and this is stored in a dictionary. A 2D matrix is used to indicate the alignment the robot needs to be in to move from one node to another. For example, graphMat[*START NODE*][*CENTRAL NODE*] will return North because the robot needs to be aligned to the North and move in this direction to reach the *CENTRAL NODE*. *pathGraph(curNode, desNode)* is a utility function that returns the path from *curNode* to *desNode*. It does this by the technique of backtracking – checking all possibilities, eliminating infeasible ones and selecting the feasible ones.

detectSims() is a function that handles the detection of the SIMs and ArUco IDs. It returns an array *nodes* which contains the AH numbers and the service required by each. **The QAH is the first element of the *nodes* array so that QAH is serviced first.**

serveAH(node) is a function that will complete the services required at *node*. As can be seen from the flowchart, a variety of other functions are used in this function:

1. *pickUp()* – picks up the supply block/trash block in front of it.
2. *placeDown()* – places the supply block in the correct place / drops trash block in TDZ.
3. *detectSupply()* – uses the PiCam to detect the colour of the supply block in front of it.

Q8. Suppose for a given arena configuration, it takes 20 seconds more to execute the task while keeping the Queen Ant Hill in priority. What will be your logic to traverse the arena in order to secure maximum marks i.e. you will serve Queen Ant Hill first by taking 20 seconds more or complete the run faster by not serving Queen Ant Hill first (Assuming, points scored for all other parameters in Total Score in both the cases remain same). Please explain and justify your logic and strategy.

(4)

The Total Score = (600-T) + (30*CSD) + (30*CSI) + (75*CSEP) + (75*CSED) + (100*QB) + (300*OB) + (DB) - (50*P)

, where the acronyms have their usual meanings as given in the rulebook. Assuming the points scored for all the other parameters in the Total Score in both the cases remain the same and is equal to *x*, and the time taken to execute the task without keeping QAH in priority is y.

Total Score for first case (keeping QAH in priority) = (600 – (y + 20)) + x + 100 = x - y + 680
Total Score for first case (not keeping QAH in priority) = (600 – y) + x = x - y + 600

Clearly, to secure maximum marks, the robot should serve QAH first and take 20 seconds more.

Challenges

Q9. What are the major challenges that you can anticipate in addressing this theme and how do you propose to tackle them?

(8)

The major challenges we can anticipate in addressing this theme and our methods to tackle them are as follows:

1. **Design of Chassis:** Chassis must hold all the components so that the bot can run smoothly. Batteries should be placed in close proximity to the motors to reduce wiring. Power bank and the microcontrollers will be kept at the other end to evenly distribute the weight and maintain the centre of gravity of the bot at the geometrical centroid of the chassis. The height of the chassis will be lowered to increase the overall stability.

2. **Construction of Pick-and-Place Mechanism:** The challenge lies in accurately halting the bot at certain distance from the blocks picking it up. Correct execution drop is also a difficult task. It may be overcome with the help of servo motors, one motor to pick and hold the block and the other one to swing the arm up and down. Arm manipulator should not obstruct the view of PiCam so it will have a mechanism to raise the arm such that it cannot hit any walls.

3. **Detection of SIMs and Blocks:** As far as this challenge is concerned, the PiCam over the bot may have some problems with angle and distance from the SIMs. To overcome this, the bot will rotate over the central node and the PiCam will be placed at an appropriate height and angle (in a manner similar to a floodlight). It will detect the blocks using shape and colour detection algorithms. This will be done using necessary OpenCV and ArUco libraries.

4. **Planning of Path Traversal:** One challenge immediately encountered is the deviation of the robot from the path. This can be avoided by appropriate use of the 3-channel line sensor. Since it is an analog sensor and 3 channels are present, the variation in values will indicate to what extent the robot has deviated from the path which can then be corrected using a suitable system, perhaps a PD controller. The other challenge is deciding which path to take when there are multiple options available. As mentioned in previous answers, the nodes are considered as vertices of a graph and appropriate path finding algorithms are applied to ensure that the correct path is taken to reach the desired node from the current location.