ROBOTIX

TECHNOLOGY ROBOTIX SOCIETY

INDIAN INSTITUTE OF TECHNOLOGY, KHARAGPUR

KSHITIJ 2016

Sherlock

Autonomous Robotics

USP: IR Hex Decoding and Heading Following

Introduction:

Sherlock is stuck in a featureless desert wasteland with nothing but a compass in his pocket. He must find his way out using the clues scattered in the desert.

Problem Statement:

Build an autonomous robot that can follow compass headings to go from start to finish using IR receiver to receive arena and waypoint information, while optimizing its path.

Event Description:

- The arena consists of multiple start and finish points. Other points are termed as waypoints. Start, finish and waypoints are collectively referred here as (points of interest: POI).

- These POIs have a **circular area of 5 cm radius** with IR transmitters on the center. The IR transmitter transmits ID of the respective POI and other information as stated below.
- The information is transmitted through messages using **NEC IR protocol**. For more information, take a look at the tutorial document.
- A message starts with "**400**" as **start code** ends with "**500**" as **end code**. "**450**" is the code used as **separator** in between the messages.
- A POI may transmit multiple such messages (max 9). Then each message will be tagged with a two digit "message_tag". **1st digit = present message id. 2nd digit = total messages to be transmitted from that POI.**
- Heading and magnetic heading as mentioned here, refer to the direction with respect to the north direction (0 degrees) that the compass would show if kept at that position. So if heading to a point is 90 degrees, it is at right angle to the point wrt north (East).

Task:

Round 1

- When the participant's run starts, the bot is placed on **start point.** The start point will provide the following information:
  - A1: Finish point for that particular run. Eg. (start msg_tag own_id finish_id stop) (400 12 0 21 500) where 0 is start ID and 21 is finish ID. **Here 400 is the start code, 12 is the message_id (1st of the 2 total messages),0 is own_id and 21 is the id of the finish point, 500 at the end being the stop code.**
  - A2: Heading for only the first possible waypoint after itself. Eg. (start msg_tag own_id [id heading cost] stop) (400 22 0 2 260 20 500). Here **400 is the start bit, 22 is the message_id (2nd of the 2 total messages), 0 is the own_id of POI and the information is that 2 is the next POI at heading of 260 degrees (from magnetic north) and cost 20, and finally 500 is the stop code.**
- All the POIs will have ids. Each POI will provide heading to a single POI. Waypoint 2 transmits the info for waypoint 4 in format as follows: (start message_tag own_id [id heading costs] stop) (400 11 2 4 260 20 500). **Here 400 is the start code, 11 is message_tag (1st of the 1 total messages), 2 is the own_id of POI, and it is connected to 4 at headings of 260 via edges of costs 20.**
- In **Round 1**, all the costs are set by default to **20.**
- The bot has to reach the final POI, which is 21 in the above example.
- The bot has to follow only the valid edges. If POI 2 gives the information that it is connected to 3 via a edge of cost 20 and heading 140, then it has to go to POI 3 only, otherwise there would be a penalty
- There will be **no explicit markings** on the arena to indicate valid edges.

- If the bot gets lost in the arena and comes back to a POI, then it has to plan a way to continue its pre planned path going through valid edges. It may take help from the walls in the arena to stumble upon a POI if it gets lost.

Round 2

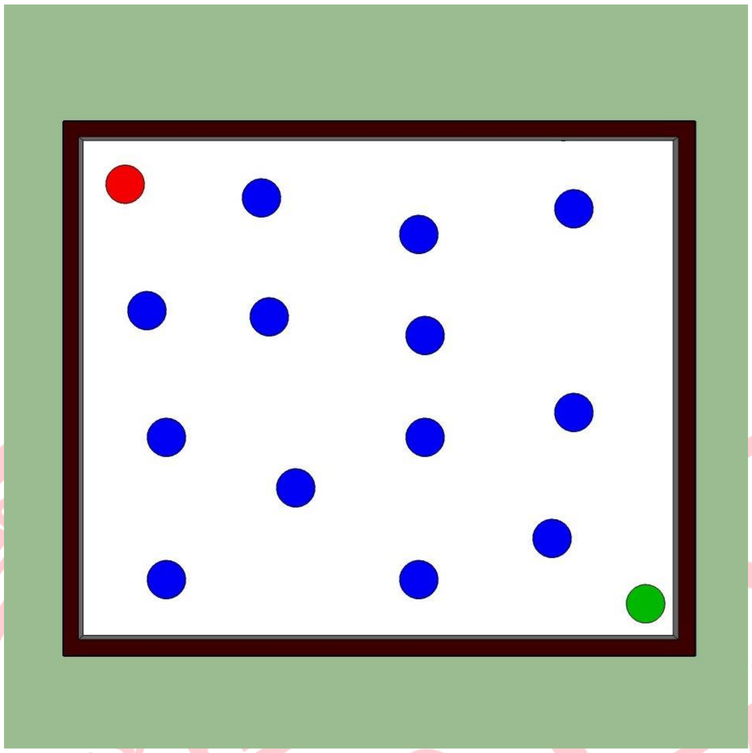
- The POIs will have ids. Each POI will provide heading to **single or multiple different other POIs**. Waypoint 3 transmits the info for waypoint 2,6,5 in format as follows: (start message_tag own_id [id heading costs] stop) (400 11 3 2 260 10 6 180 45 5 300 39 500). Here 400 is the start code, 11 is message_tag (1st of the 1 total messages), 3 is the own_id of POI, and it is connected to 2,6 and 5 at headings of 260, 180 and 300 via edges of costs 10,45 and 39 respectively.
- When the participant's run starts, the bot is placed on start point. The start point will provide the following information:
  - A1:
  - **POIs each waypoint directs to.** Eg (start msg_tag own_id [waypoint [waypoint_it_directs to]s separator]s stop) (400 13 0 2 3 6 5 450 3 5 2 7 450 500). **Here 400 is the start bit, 13 is the message_id (1st of the 3 total messages), 0 is the own_id of POI and the information is that 2 is connected to 3,6 and 5, 3 is connected to 5,2 and 7, and finally 500 is the stop code. The 450s as mentioned are separators.**
  - A2:
  - **Finish point for that particular run.** Eg. (start msg_tag own_id finish_id stop) (400 23 0 21 500) where 0 is start ID and 21 is finish ID. **Here 400 is the start code, 23 is the message_id(2nd of the 3 total messages),0 is own_id and 21 is the id of the finish point, 500 at the end being the stop code.**
  - A3:
  - **Heading for only the first possible waypoint after itself.** Eg. (start msg_tag own_id [id heading cost] stop) (400 33 0 2 260 25 500). **Here 400 is the start bit, 33 is the message_id (3rd of the 3 total messages), 0 is the own_id of POI and the information is that 2 is the next POI at heading of 260 and cost 25, and finally 500 is the stop code.**
- So the bot gets to know the entire connected graph on the start POI.
- The bot has to plan its route passing through the minimum number of POIs from its start and minimising the cost of travel along its way to reach a predefined finish point while, traversing through only the **valid edges**. Valid edge means, if POI 1, publishes heading for 2, 3, 6. Then it should go to any of the three only, **otherwise there will be a penalty.**

- The bot has to minimize the score given by **A x (number of POIs traversed) + B x cost**. A and B will be given on day 1 of the event. The final score would be as described in scoring below, which would be used to evaluate the winner.
- There will be no explicit markings on the arena to indicate valid edges.
- The cost between two edges is a **biquadratic function of the absolute value of their difference in their IDs modulo 5**. The cost between POIs with ids m,n is defined thus as:
    - **cost(m,n) = a x^4 + b x^3 + c x^2 + d x + e , where x is abs(m-n)%5 and % is the modulo operator as in C/C++, denoting the remainder when absolute value of m-n is divided by 5, which can be 0,1,2,3 or 4. Please note that the difference between m and n is m-n or n-m respectively if m is larger than n or n is larger than m.**
    - Thus cost of an edge between POI 2 and 16 is 256a+64b+16c+4d+e, as (16-2)%5 is 4. Cost between 0 and 1 would be a+b+c+d+e, as (1-0)%5=1
    - All the constants a,b,c,d,e are all positive integers. **The a,b mentioned are different from the A,B as mentioned earlier.**
    - The constants a,b,c,d,e are unknown and the bot has to find them itself using techniques of simultaneous equations. The robot knows the edge IDs and can solve five equations after getting sufficient information.
    - Please note that this biquadratic edge cost term comes into play only in round 2, round 1 has all edge costs are 20.
- If the bot gets lost in the arena and comes back to a POI, then it has to plan a way to continue its pre planned path going through valid edges. It may take help from the walls in the arena to stumble upon a POI if it gets lost.

Arena:

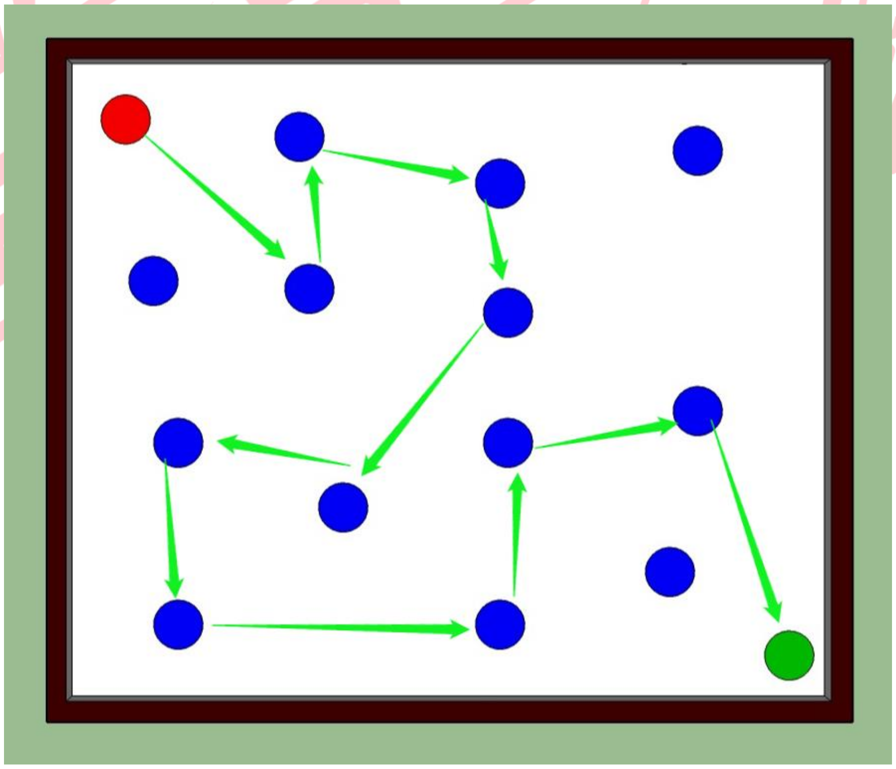
Depictive Arena

- Legend:
    - Red POI: Start point
    - Blue POI: Waypoints
    - Green POI: Finish point
    - Green Lines: Valid edges

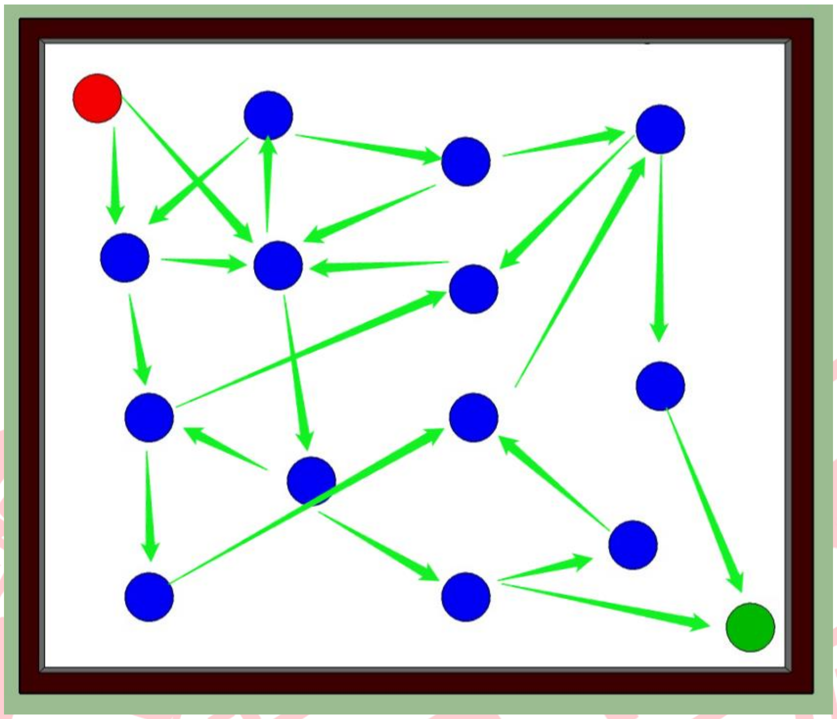


Sample Arenas

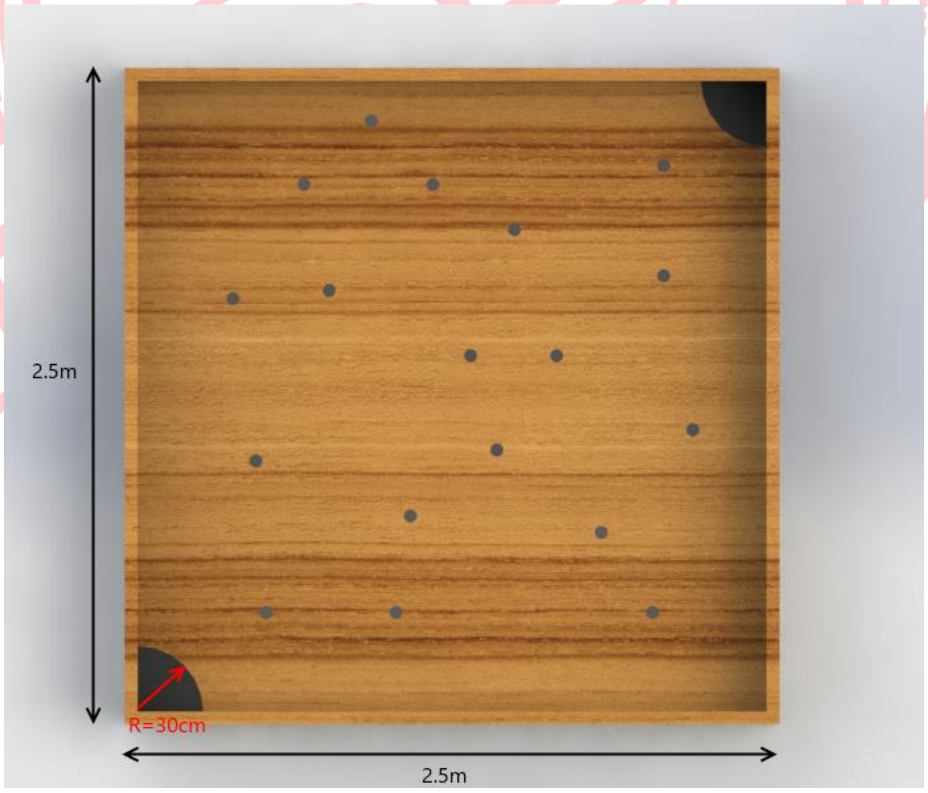
Round 1



Round 2



Actual Arena



Specifications:

Round 1

- Each waypoint will send the bot to **only one** other waypoint
- The bot must simply follow the directions to the end point
- The round will be of **3 minutes duration**, after which the run would be stopped immediately and score would be calculated.
- If finish point is reached then the bot has to **flash a LED**
- There is **1 timeout** and **1 restart** allowed.

Round 2

- Waypoints will be transmitting heading and cost information for **single or multiple other waypoints.**
- Start points will transmit the heading and cost for its immediate neighbour POIs and providing referencing information for the entire arena, i.e. which waypoints each waypoint on the arena provides the headings for.
- Robot has to plan its best optimized path with the information being given from the start point as mentioned in "Task" above.
- Once the bot reaches the finish point it must **flash a LED.**
- The round will be of **5 minutes duration**, after which the run would be stopped immediately and score would be calculated.
- There are **2 timeouts and 1 restart** allowed.

Rules and Specifications:

General Rules

- The robot should be completely autonomous.
- All arena dimensions have a tolerance of 10%.
- Each team can consist of a **maximum of 4 members.**
- Each team should have unique participants i.e. no two teams can have even a single participant common.
- The team members can be from different institutes or colleges.
- The right spirit of participation is expected from the participants.
- **The decision of the Team ROBOTIX will be final and binding.**

Technical Rules

- A standard **220 volt AC supply** will be provided by Team Robotix in the arena.
- All circuitry and sensory equipment should be placed on the robot adhering to the ROBOT SPECIFICATIONS.

- Participants will have to bring their own programmers, cables and softwares. No programmers will be supplied.
- Hard coding is NOT allowed. Hard coding is defined here.
- The robot can be powered on-board as well as off-board.
- No kind of external control will be allowed.

Robot Specifications

- Each robot can have maximum dimension of **25x25*x*20 cm3 (LxB*x*H)** respectively.
- No part/mechanism of/on the robot should exceed the given dimensions before the commencement of the event run. The robots can exceed their respective dimensions once the event commences.
- The autonomous robots should be on-board processing robots, i.e., the robots cannot be controlled by a remotely kept computer.
- LEGO kits or its spare parts or pre-made mechanical parts are not allowed.
- Ready-made gearboxes, sensors, development boards can be used but no other part of the robot should contain any ready-made components. Simple car bases with no extra features may be used.
- The bots should not harm the **Sherlock** event arena in any way. If it does so, a penalty will be imposed on the team. The magnitude of the penalty will be decided by **Team ROBOTIX.**
- Processors of **less than 16-bits are allowed**. ARM processors are not allowed.

Event Rules:

Robot:

- The robot should be completely autonomous.
- The robot needs to traverse a **2.5m x 2.5m** arena for both rounds
- The arena has multiple number of IR LEDs (POIs) that transmit information according the NEC IR Protocol.

POIs:

- The POIs are small holes in the arena that have an IR LED inserted into each of them. The range is such that the TSOP1738 IR Receiver can detect a POI from a distance of about 2 cm from the POI.
- The POIs would transmit binary data at **38KHz frequency in strictly NEC IR format.** The format as detailed in the tutorial on the website.

Magnetic Heading:

- The magnetic heading would strictly be calculated with respect to north pole of earth wrt IIT Kharagpur campus. Although, the code doesn't require calibration as long as it is written anywhere in India/neighbours
- We have used Adafruit HMC5883L library for calibration. Usage of other compass modules or other libraries is not recommended and any discrepancies thus caused would not be the responsibility of Team Robotix.
- The Magnetic Heading would be given in **degrees** from 0 to 359 and **not in radians.**

Restarts/Timeouts:

- A maximum of **1 Timeout of 2 minutes each in round 1 and 2 timeouts in round 2** may be taken. Penalty will be imposed for each timeout and robot will start from the last node crossed.
- The participant's robots can have a maximum of **1 restarts**. A penalty will be imposed on the team for every restart that they take.
- In case of a restart the participant's robots will be set to their initial positions. Timer will be set to zero and the run will start afresh with the addition of the penalty for restart.
- If the robot goes off track or outside the arena, then the team has to take a restart or call off their run.

Scoring:

- Base Score: **S**
- Reach Finish Point: **500**
- Reach each POI: **-A (**as mentioned description)
- Traversing an invalid edge each time: **-100 (X)**
- Total Cost incurred in reaching the end point: **TC**
- For each timeout: **-150 (T)**
- Restart: **-250 (R)**

Scoring Formula:

- **Score = S- A x (no of nodes traversed)- B x (TC)-Penalties (X,T,R)**
    - So if a participant reaches the end point traversing 9 nodes, with 250 total cost incurred and with 2 invalid edge traversals, 1 timeout and 1 restart, his score would be:
    - Score = S-(9 x A)-(250 x B)-(2 x 100)-(1 x 150)-(1 x 250)+500
- Target is to **maximize the score.** If two participants manage to get the same score, then the participant with lowest time taken is declared the winner.

Tutorial & Resources

Visit www.robotix.in/event/sherlock to check out the latest Event Updates.

Read our Tutorial for Sherlock.

Join the Event Facebook Group for latest updates and doubt sessions.

Contact

For queries, contact our Event Heads:

Aditya Narayan	Riya Bubna	Mratunjay Gupta
aditya.narayan@robotix.in	riya@robotix.in	mratunjay@robotix.in
+91-8609283200	+91-9007388802	+91-8609266778