CS 666 – Mobile Robotics

Term Project

Travelling Salesman Problem

Under the guidance of :

Dr. S. B. Nair

Presented at:

Indian Institute of Technology, Guwahati

Report by Group 5 –

Vinayak Jadhav (154101017)

Sahil Manchanda (154101019)

Manasi Sant (154101022)

Prateek Yadav (154101028)

Problem Statement:

Travelling Salesman Problem with multiple salesman as robots.

Mapping of TSP to real world problem :

In this project we converted TSP problem into demand supply system. TSP problem has a graph in which node is a city. In our case we are treating node as a farmer's place. Farmer's place has ability to generate demands. This demand will indicate amount of vegetables produced by the farmer. Each farmer's place (node) has a computer (device which Is used to communicate with neighbouring nodes) with it. When demand comes, farmer will submit the demand into the computer.

There will be multiple robots running in the environment. Robot can be visualized as truck (transportation unit). Our objective is to fulfill all the generated demands at the farmer's place in best possible time. We are targeting decentralized system. Every robot will take decision on its own.

Assumptions:

- Robot will not physically sense the demand.
- When robot is fulfilling any demand its not physically carrying any load from source to destination.
- When node generates a demand it will not generate another one till previous is fulfilled.

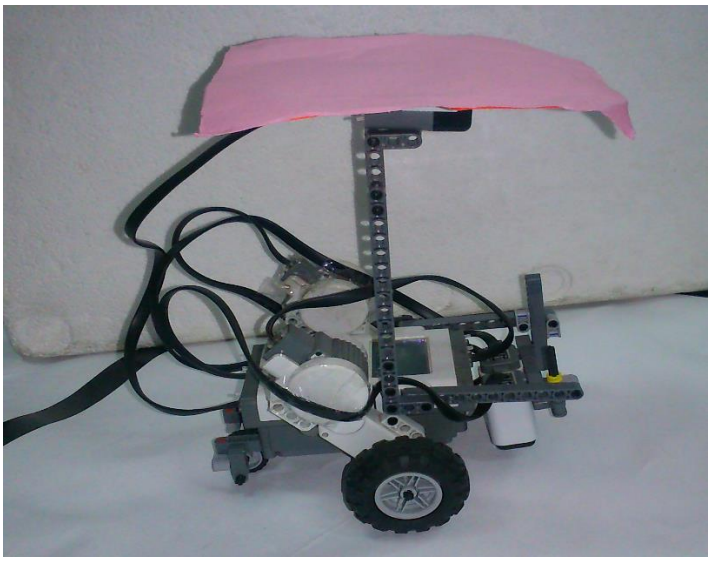
Technologies used:

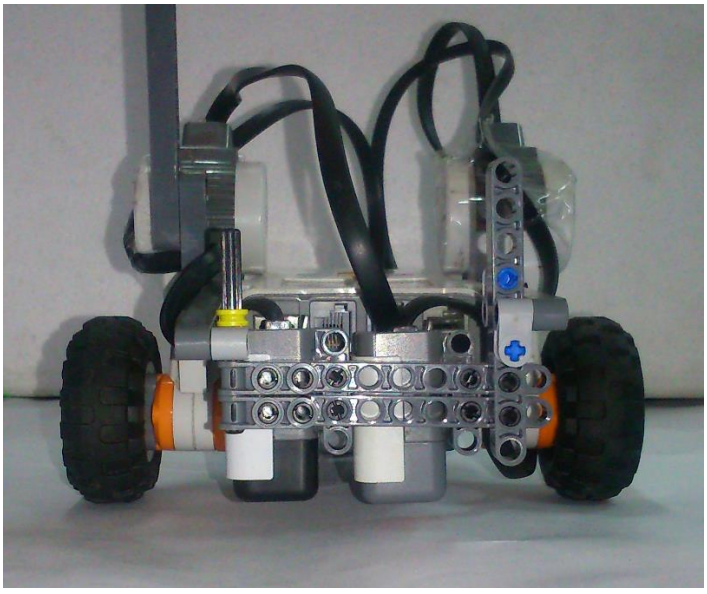
- Software
    - 32 bit jdk with version 8
    - 32 bit eclipse (Luna or above)

- Hardware
    - Lego NXT kit
    - Bluetooth connectivity for communicating with NXT Brick

Robot Configuration:

- Each robot has three sensors :
    - Light Sensor - Required for black line following.
    - Color Sensor - Required for Node detection and also for taking turns (LEFT, RIGHT, STRAIGHT, U-TURN) at nodes and junctions.
    - Compass Sensor – Required to align the robot to a particular direction while changing path.
- A rectangular colored paper (unique color for each robot) of size (15cm x 15cm) is placed on top of each robot. This colored paper is required to find the position of robot from overhead camera.

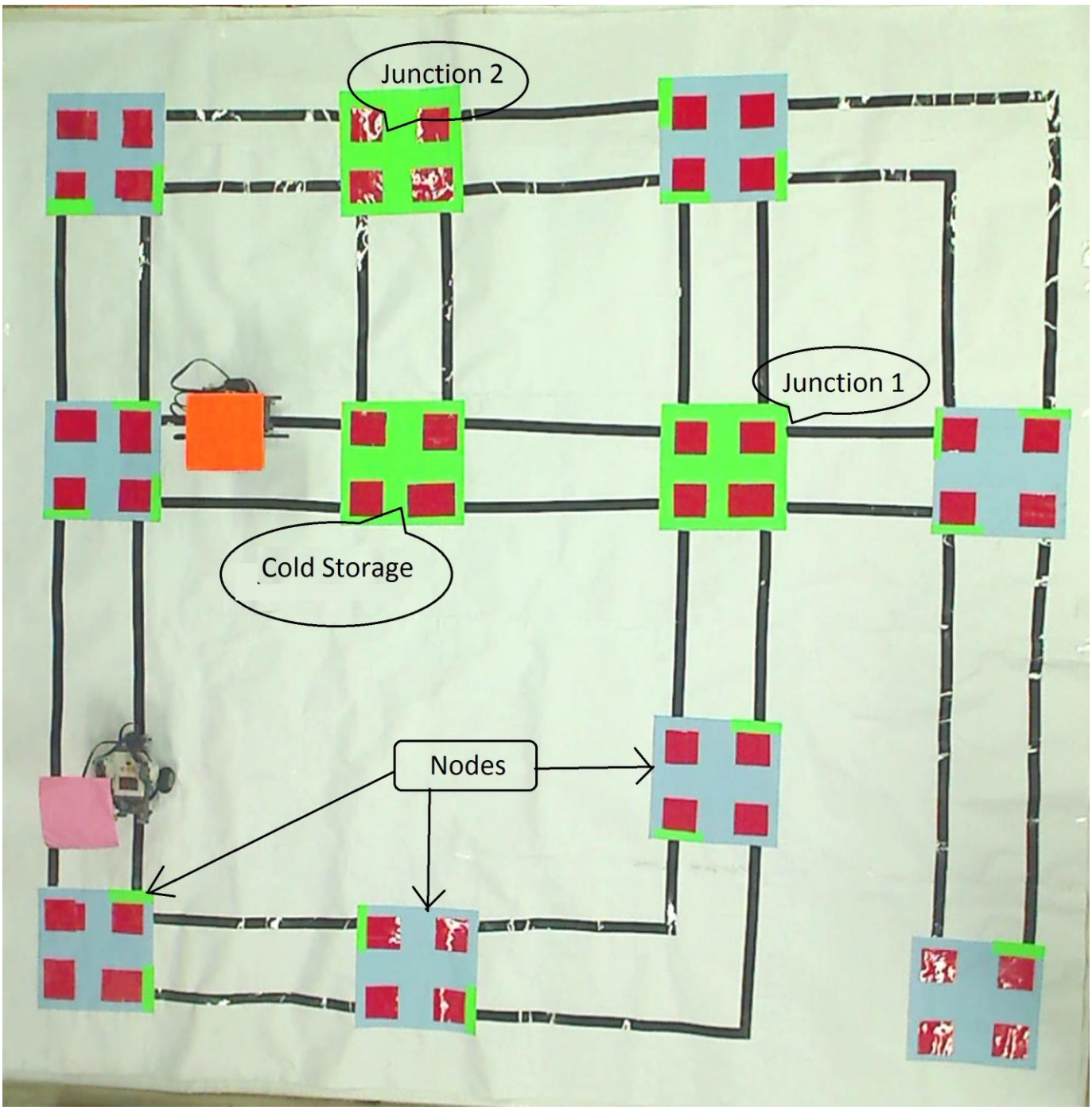




Map Design:

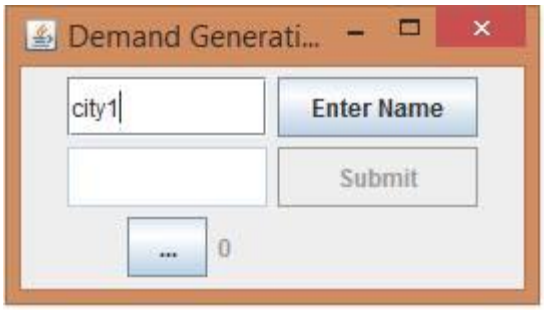
In the current map we have 8 nodes and 2 junction nodes. One cold storage node is there.

Nodes are connected by black lines. It is a two-way path. Junction is a node where 3 or more paths are meeting.

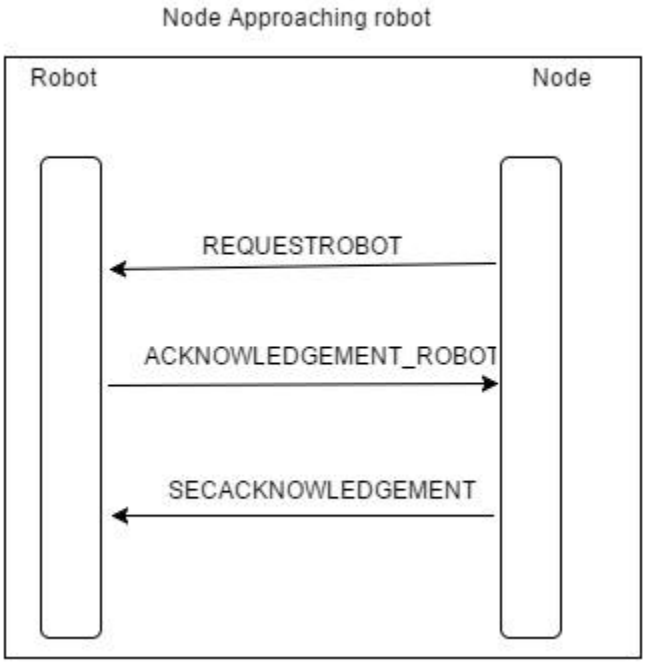


Algorithm (Node)

Node generating demands



1) When a Node generates a demand, first it searches for robots in its vicinity for a certain period of time.
2) If the Node finds a robot in its vicinity, it requests the robot by telling amount of load to be carried and waits for ACK (Positive/Negative) from robot.
3) If there are multiple robots in the vicinity of Node, then requests are made in increasing order of distances. As soon as a robot replies with a Positive ACK, no further requests are made to other robots in the vicinity.
4) If there is a Positive ACK from a robot, then a confirmation is sent from the Node to the robot. This establishes 3-way handshake.
5) If there is no Positive ACK from the robots in the vicinity or there is no robot in the vicinity, then the request is passed to the neighbouring nodes.



Robot physically reaches a node

1) If a robot reaches a Node, the Node will receive a request from robot asking the current demands at the Node.
2) The Node will reply to this request with its demands as well as its neighbour's demands.
3) The robot will send Positive/Negative ACK for each demands to the Node.
4) For every Positive ACK from robot, the current Node will inform the source Node of the demand that a robot is available to fulfill their demand, the current Node will wait for responses from all the source neighbouring nodes.
5) The current Node on which robot has arrived, will send a confirmation to the robot for all the neighbouring source nodes which have agreed for this robot. This is done after getting responses from all the source neighbouring nodes which have demands.
6) If a Node receives a signal from its neighbouring nodes that a robot is available for it and still this Node's demand is not fulfilled yet, then it will say YES for the robot, else NO. Also it will inform its neighbours to stop searching for robots for it.
7) If the Node receives a REQUESTFULFILLED signal, then it will mark the demand as fulfilled.

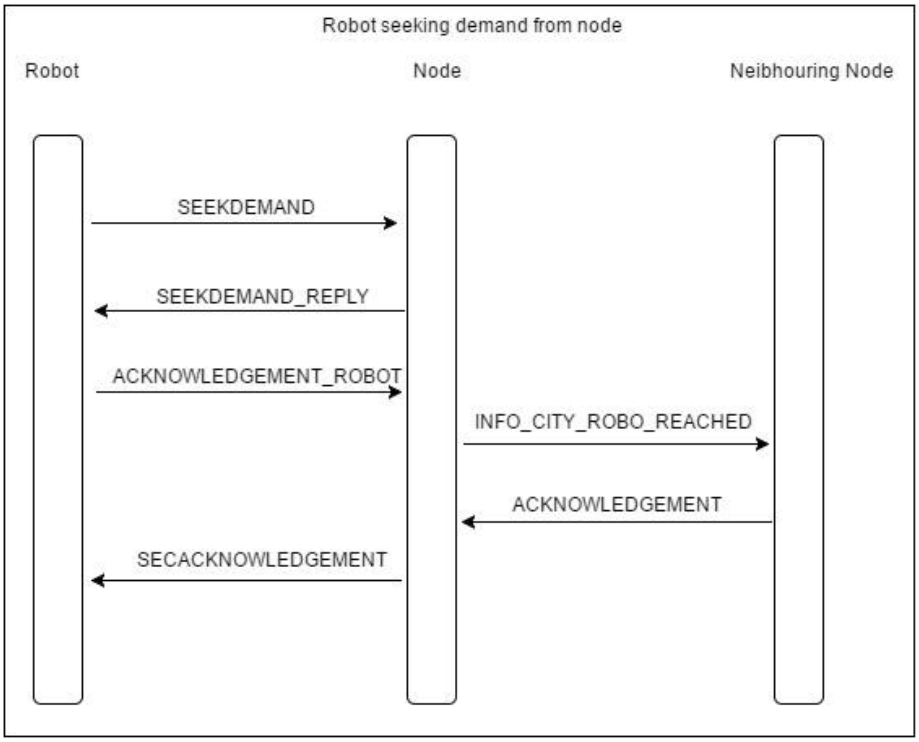
Algorithm (Robot side):

When robot will receive request from node (REQUEST_ROBOT signal):

Robot has to take decision whether to serve this request or not. This request might be from the same node or from its neighbour.

When robot reaches node :

1) It will check if this is the node which is demanding. It will fulfil this demand at the current node. Capacity will be increased accordingly.
2) Robot will ask node if it has any demands pending or not (SEEK_DEMAND signal). It will get some requests (node demand or neighbouring node demands).
3) For each request:
    a. Robot will take decision whether it can fulfil it or not.



When robot reaches cold storage:

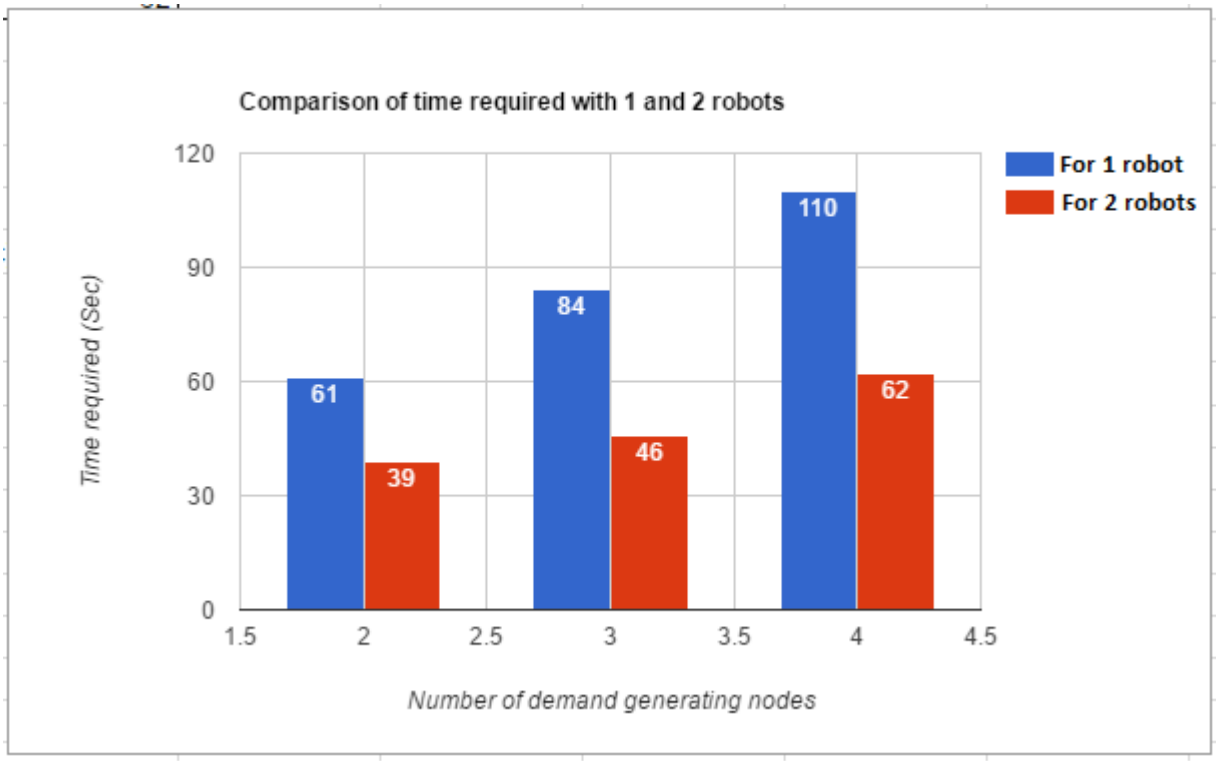
1) Robot has completed all demands. Now robot capacity is 0.
2) Currently it has no requests to be served. Now it will scan through all the completed requests till now.
3) For each node in history :
    a. Calculate maximum demand by the node
4) If maximum demand is at node which is far from cold storage (current position) then go for second best.

Robot decision making procedure:

1) Robot will check if it can fulfil the new request after doing all existing ones in terms of capacity.
2) If it can serve the request in terms of capacity then,
    a. It will choose the shortest path to cold storage covering all existing requests including this new one.
    b. If path cost of this calculated path is not too high (heuristic used is not greater than currentPathCost * 2) then
        i. Robot will accept the request. Declare this decision to requesting node by sending acknowledge.

ii. Add this new request to array of requests which it is going to fulfil.
iii. It will choose next destination node according to new path to be followed.

Observation:



With 2 robots the net lateness of the system is reduced considerably.

Issues:

- Color Sensor – Color sensor was giving undesirable results which leads to wrong decisions by robot.
- Motor – When equal power is given to 2 motors, rpm observed was not equal.

Future Enhancements:

- This decentralize approach can be extended to hybrid model of centralized and decentralized.
- Whenever decentralize approach will fail (demand is waiting to be fulfilled for long time) then centralized approach will take over.
- Robot must physically sense the demand when it reaches the node.