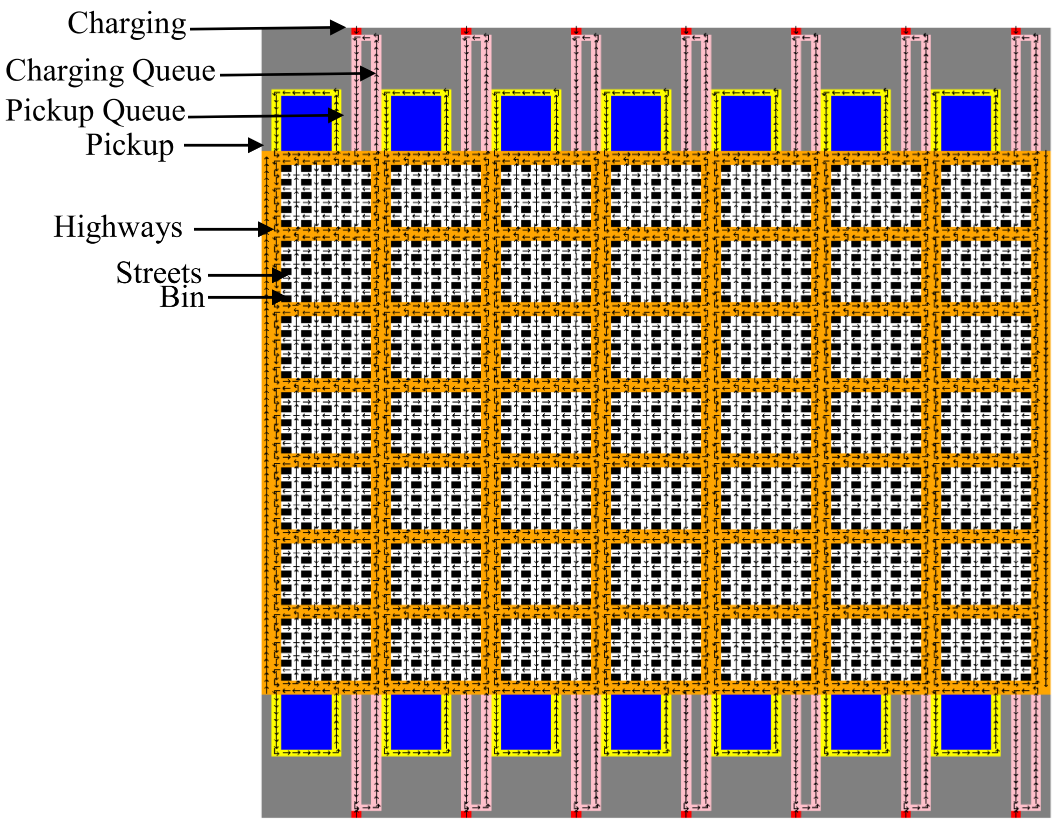
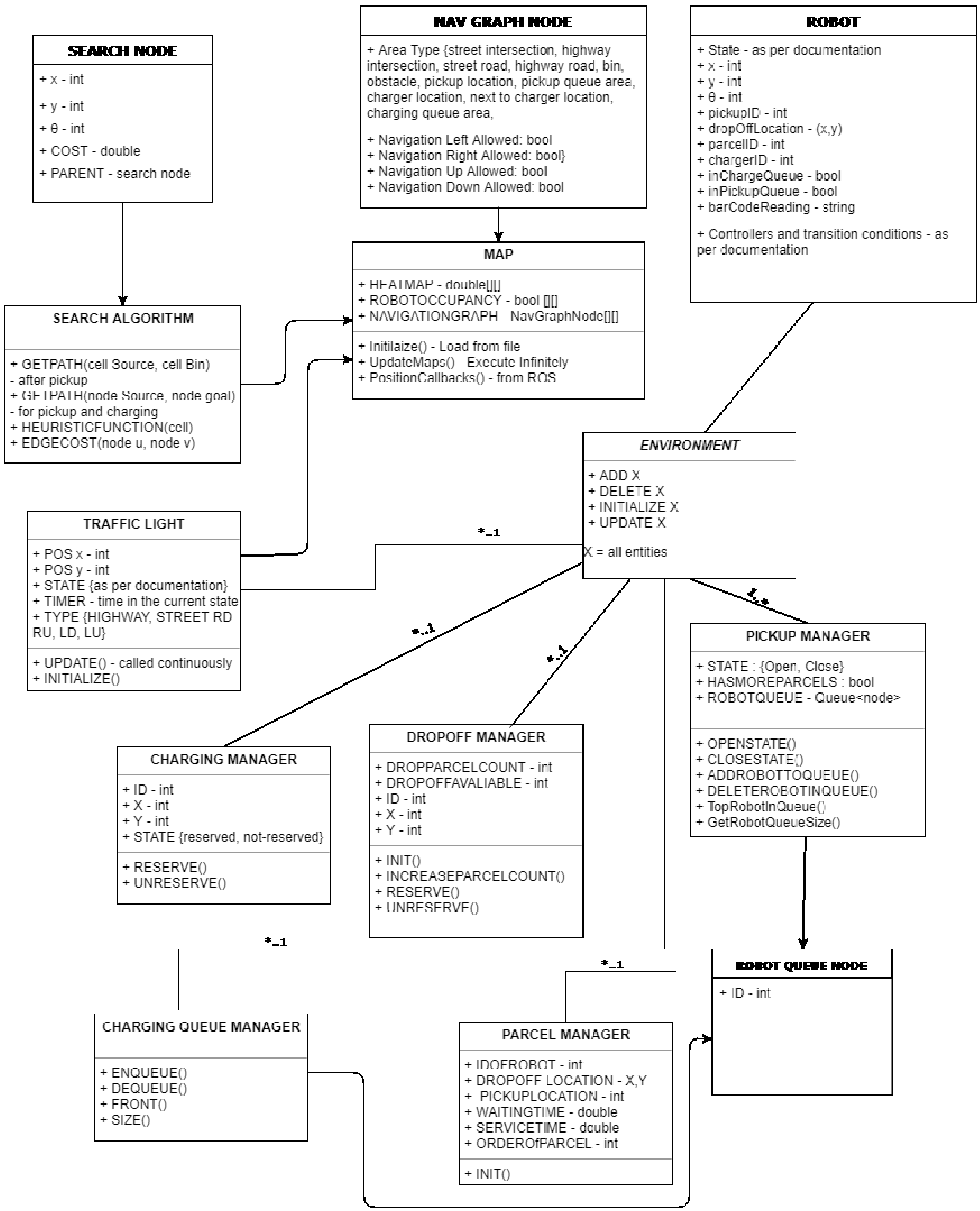
Sorting Robot

Sorting Robot sorts the parcel into their respective bins depending on the locations using barcodes. Map for sorting robots includes parcel bin, streets and highways for robot movements, robot charging points, parcel pickup points where human will place parcel conveyor belt to robot top. Once parcel is placed on robot top, robot should place the parcel to its respective bin by following best possible path and avoiding the congestion as much as possible.

Navigation Graph



CLASS DIAGRAM:



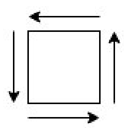
Class Diagram

Heat Map: Heatmap is used for conges tion avoidance. Search Algorithm avoids area of high heat value. A star algorithm tries to schedule robot to go through area of lower cost thus distributing map to avoid congestion at particular place.

HeatMap(t)= ηΔt HeatMap(t-Δt)+ OccupancyMap(t)
η = evaporation rate (typically 0.75 per millisecond)

Search:
A* Algorithm. It operates on (x,y,θ) as state vector.
There are two variants of the algorithm:

1. Shortest cost path to a bin in which case destination can be anyone of four adjoining edges.



2. It is the case of charging /pickup queue, where goal is available (x_G, y_G, θ_G) representing location of starting of the queue of chosen charging station / pickup point.

a. Edge Cost:

$$w(u < u_x, u_y, u_\theta >, v < v_x, v, v_\theta >) = \left(\frac{|u_x - v_x| + |u_y - v_y|}{v} + \frac{angleDifference(u_\theta, v_\theta)}{\omega} penalty\right)(1 + \alpha\ heatMap(v_x, v_y))$$

v=maximum linear speed, ω=maximum angular speed, penalty is applied to penalize stopping and turning (≈1.2), α penalizes heat map (to experiment)

b. Heuristic Cost for bin search:

$$h(u < u_x, u_y, u_\theta >) = \max\left(\frac{|u_x - G_x| + |u_y - G_y|}{v} - \frac{1}{v}, 0\right)$$

Here (G_x,G_y) stands for the bin location

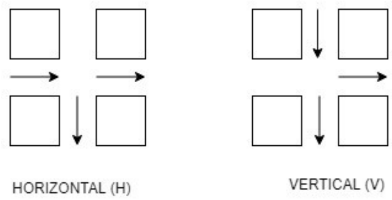
c. Heuristic Cost for goal search:

$$h(u < u_x, u_y, u_\theta >) = \frac{|u_x - G_x| + |u_y - G_y|}{v} + \frac{angleDifference(u_\theta, G_\theta)}{\omega} penalty$$

Traffic Light
Purpose of Traffic Light Management System is to ensure the robots don't collide **sideways** which is important given that there is no proximity sensor that tells robots on the sides.

The intersection management policy is that at any point of time only one robot will be allowed to center the intersection thus avoiding collision. The traffic light ensures adherence of the policy.

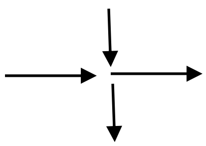
1) Traffic light related to the street intersection has 2 states:
   a) Traffic originating from Horizontal Lane can flow
   b) Traffic originating from Vertical Lane can flow

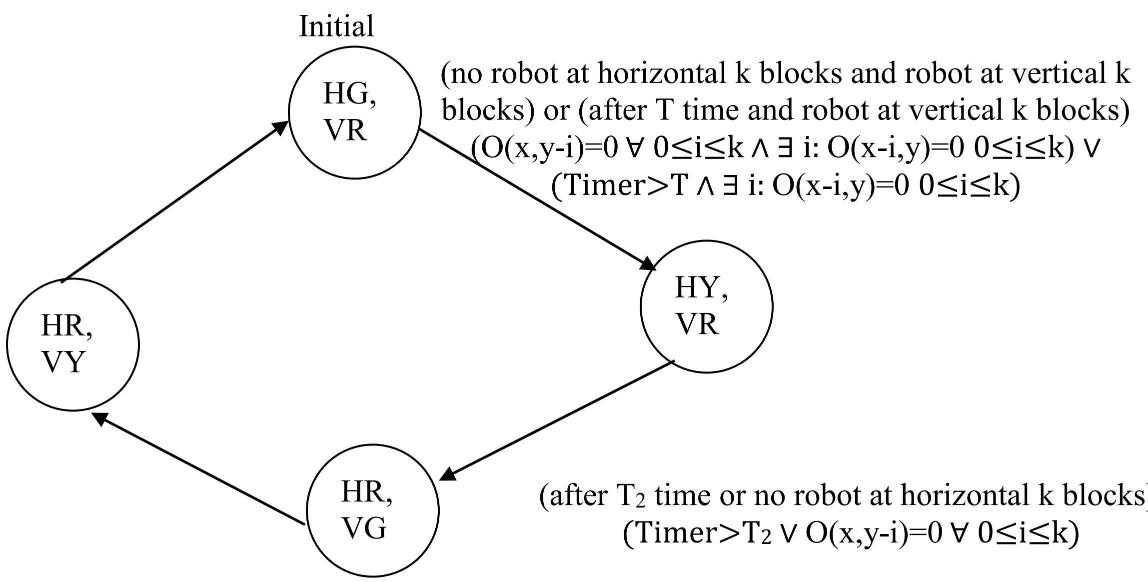


Traffic light change is modeled as FSM where vertices represent state of traffic light and the edges represent condition meeting which the state will change.

The FSM representing Traffic Light change is given below:

Case RD (right down):

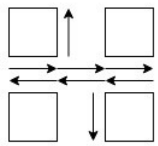
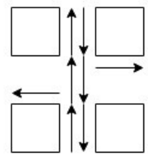




HG, VR: Horizontal Green/Vertical Red
HY, VR: Horizontal Yellow, Vertical Red
HR, VG: Horizontal Red, Vertical Green
HR, VY: Horizontal Red, Vertical Yellow
RD : Right Down
O : Occupancy Map

Cases Right Up (RU), Left Down (LD), Left Up (LU) can be similarly enumerated.

2) Traffic Light for Highways: It operates in two states:

Here also Traffic Light condition is exactly same as for Street Traffic Lights.

Coordinate Axis Systems

The navigation map, heat map, search and traffic light use the cell coordinate axis system where origin is top-left, X axis is the vertical axis, Y axis is the horizontal axis and the coordinate system is discretized into cells.

The robot positions and simulator use the real coordinate axis system.

A central utility exists to convert the world coordinate axis system into cell coordinate axis system.

Robot BFSM:
Overall Architecture to control Robot a behavioral FSM. There vertices represent different types of controllers. Robot at any state means the corresponding controller is loaded on the robot for motion. The edges represent the transition condition which are continuously monitored. Upon reaching the transition condition the controller is changed.

Behaviors:
Select Pickup Behaviour
pickup = arg min w_1 CostToReachPickup(i) + w_2 PickupQueueSize(i) : pickup(i) has more parcels

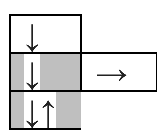
Select Charger
pickup = arg min w_1 CostToReachCgarger(i) + w_2 ChargerQueueSize(i)

Robot Controller Callbacks to Ensure:
i) Callback when robot enters the pickup area, if operational (robotInPickupQueue = False and RobotInPickupArea=true and Pickup(i) has more parcels). If the condition is true, call "enqueue pickup queue" and set robotInPickupQueue = True

ii) Callback when robot enters the charging area (robotInChargeQueue = False and RobotInChargeArea=true). If the condition is true, call "enqueue charge queue" and set robotInChargeQueue = True

iii) Callback when robot leaves the pickup area (pickup cell only) (robotInPickupQueue = True and RobotInPickupArea=false and Pickup(i) has more parcels). If the condition is true, call "dequeue pickup queue", set robotInPickupQueue = False, set pickup state = open

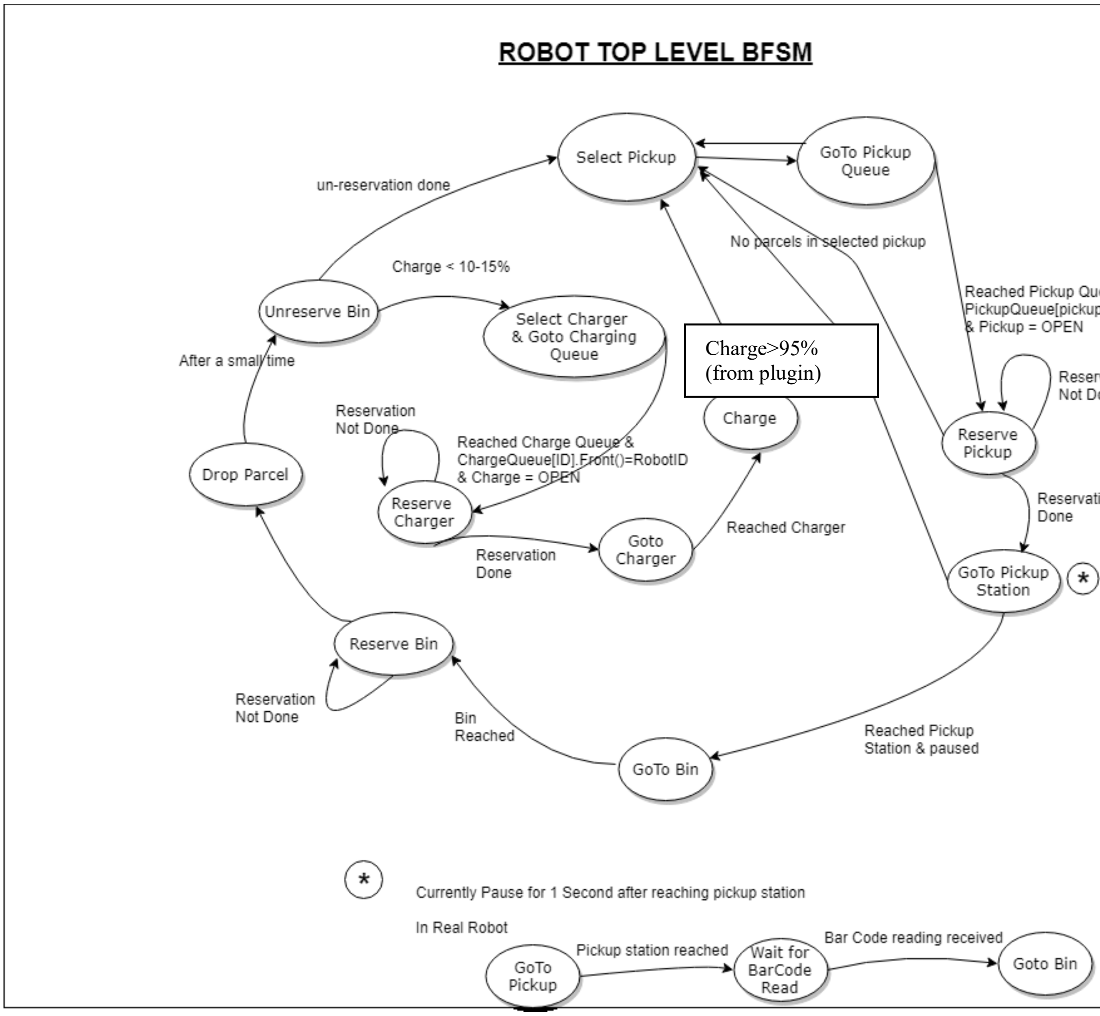
iv) Callback when robot leaves the charge area (charge cell and the next cell) (robotInChargeQueue = True and RobotInChargeArea=false). If the condition is true, call "dequeue charge queue", set robotInChargeQueue = False, set charger state = open



Charging Point

v) Callback when the robot is in a pickup queue, but the pickup queue has no parcels. Change state as per BFSM.

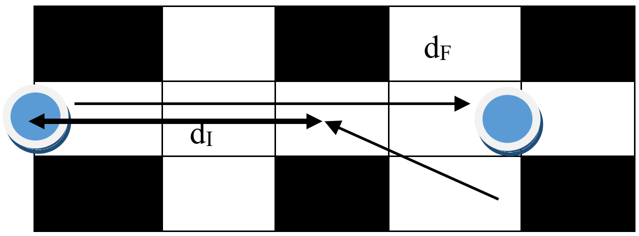
In all cases, the information from area is available from the annotated navigation map and the charger/pickup ID selected by the robot


ROBOT TOP LEVEL BFSM
Select Pickup
GoTo Pickup Queue
un-reservation done
No parcels in selected pickup
Charge < 10-15%
Unreserve Bin
Select Charger & Goto Charging Queue
Charge>95% (from plugin)
Reached Pickup Queue & PickupQueue[pickupID].Front()=RobotID & Pickup = OPEN
After a small time
Reservation Not Done
Charge
Reserve Pickup
Reached Charge Queue & ChargeQueue[ID].Front()=RobotID & Charge = OPEN
Drop Parcel
Reserve Charger
Reservation Done
Goto Charger
Reached Charger
GoTo Pickup Station
*
Reserve Bin
Reservation Not Done
Bin Reached
GoTo Bin
Reached Pickup Station & paused
*
Currently Pause for 1 Second after reaching pickup station
In Real Robot
GoTo Pickup
Pickup station reached
Wait for BarCode Read
Bar Code reading received
Goto Bin

Robot "Goto Goal" Behavior

Level 1a: Goal is forward ahead a certain distance

Assume that the robot senses d_F from the front proximity sensor. The cell corresponding to the physical location of the robot at front can be calculated. The robot must never stop/aim to stop at an intersection. Therefore, before d_F, the first non-intersection road is calculated. Let it be at a distance of d_I. This is only when the behavior is non-turn. Note that, unlike previously, the world coordinate axis system is now in place.



Bins

Last Non-Intersection

$v=K_P \min(d(q,goal), d_I)$
$\theta_G=\text{atan2}(G_y-y, G_x-x)$
$\omega= K_P \text{ angleDifference}(\theta,\theta_G)$

$d(q,goal)$ = Eucledian distance between current position and sub-sub-goal
d_F= distance from front proximity sensor
d_I = distance to the previous intersection with no robot

Level 1b: Goal is at the same place with a $\pi/2$ orientation

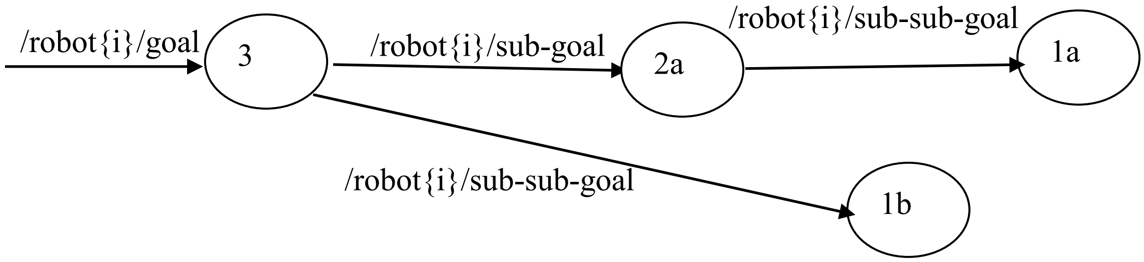
$v=K_P (d(q,goal))$: iff goal ahead of robot
θ_G=desired goal angle
$\omega= K_P \text{ angleDifference}(\theta,\theta_G)$

Level 2a: Traffic Light Aware Motion Planner

Read the current robot position and current a sub-goal directly ahead
Set sub-sub-goal in the path as the last point where traffic light is yellow/red or last point where a turn needs to be performed.
Publish sub-sub-goal

Level 3: Sequencer

Get path from search algorithm
Simply the path as sequences of straight motion and turn only
(e.g. (1,1), (2,1), (3,1), (4,1), (4,2), (4,3), (4,4) becomes (1,1) (4,1) turn (4,4))
while path is not null
    select sub-goal as next turn/straight path at a maximum distance k
    publish sub-goal and do not wait for action server
    plan a new path from sub-goal
    simply the path as sequences of straight motion and turn only
    wait for action server



RQT-GRAPH: Rqt Graph showing nodes, topics and date to be passed between nodes is shown below: