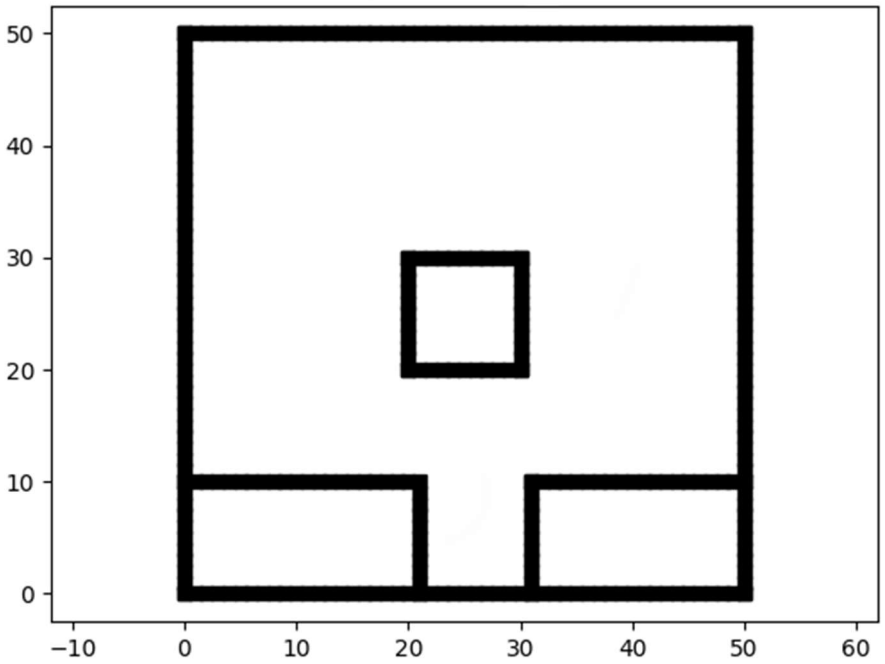
Valet Parking

In this report, we describe our approach used to parallel park a diwheel robot, a car and a truck with a trailer in a given obstacle field. Our obstacle field can be seen below.



Here the center square is the main obstacle while the lower left and right boxes can be assumed to be two parked cars/obstructions. Our test vehicle will start at the top left (North West) part of the map and will try to park itself in between the 2 obstructions shown in the bottom(South) of the map.

Diwheel Robot

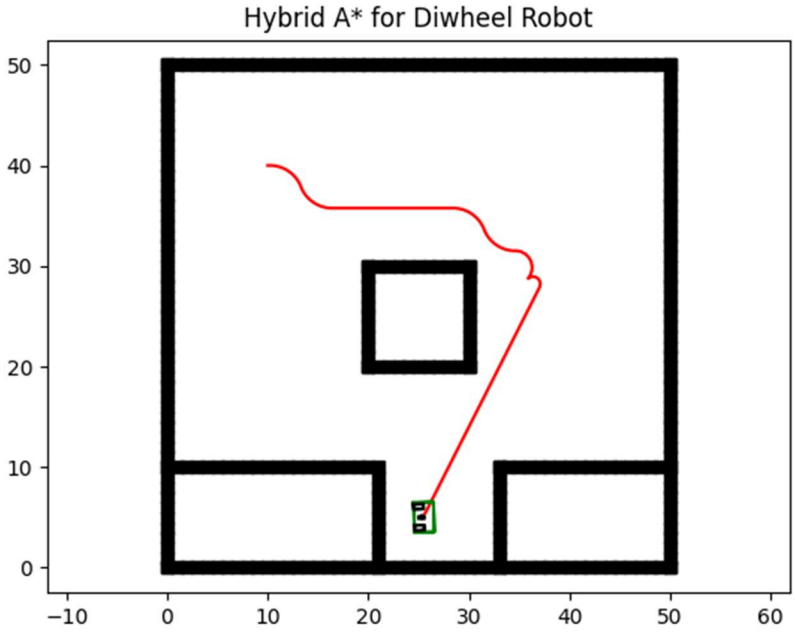
Starting off with a diwheel robot, we apply the kinematic model i.e. skid steer of a diwheel bot in our algorithm to traverse and reach the goal position and orientation. This gives us the ability to rotate at a point, the path traced out by our algorithm can be seen in the figures below. Kinematic equations used:

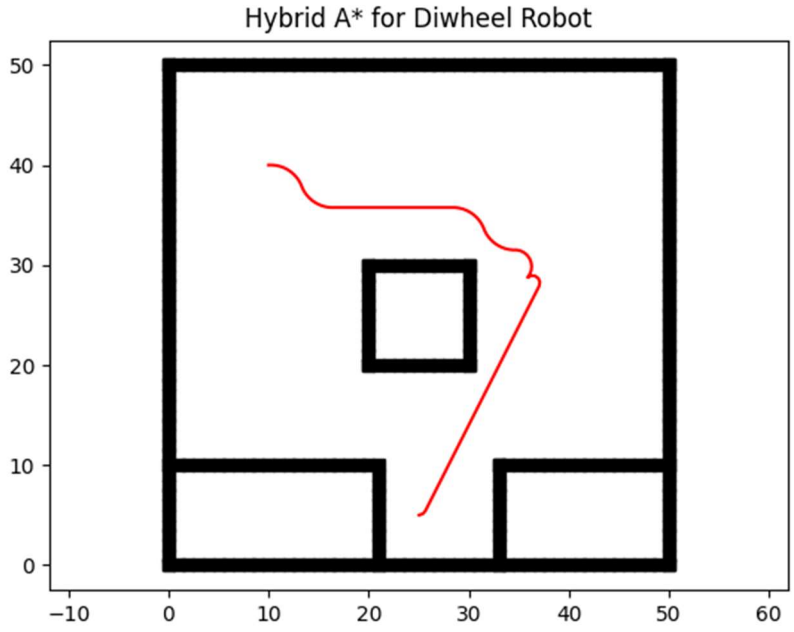
$$v = \frac{R}{L}(v_R + v_L)$$
$$\omega = \frac{R}{L}(v_R - v_L)$$

where R is the wheel radius and L is distance between the 2 wheels.
The position (x, y) and yaw ϕ:

$$\dot{x} = v\cos\phi$$
$$\dot{y} = v\sin\phi$$
$$\dot{\phi} = \omega$$

The constraint equation is $\dot{x}\sin\phi - \dot{y}\cos\phi = 0$.





The above two images show the path traversed with and without the test subject i.e. diwheel robot. The “3” we can see in the drawn out path is due to the use of a forgiving reversing cost in our algorithm. We also use reeds-shepp curves for a smoother trajectory at the end. More about

reeds-shepp curve can be found in the appendix of this report as we will be using it for all of our test subjects.

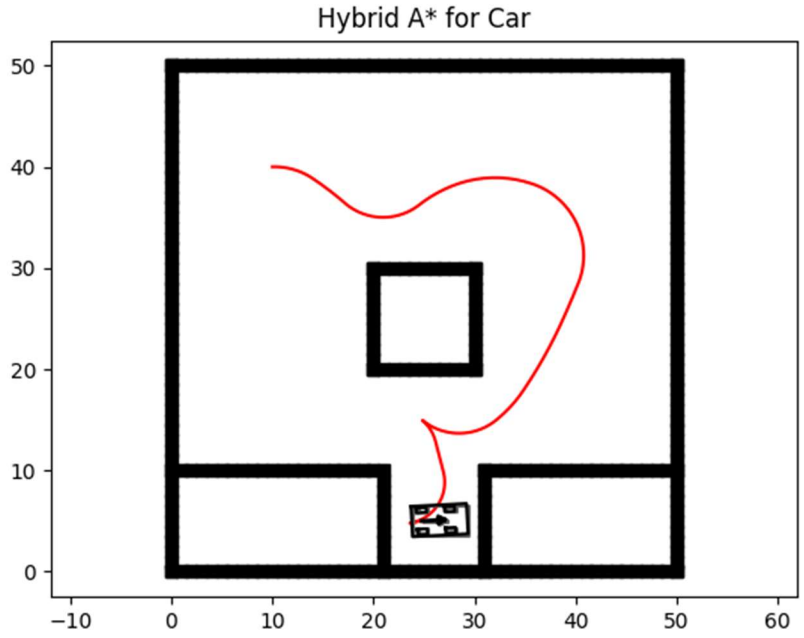
Car

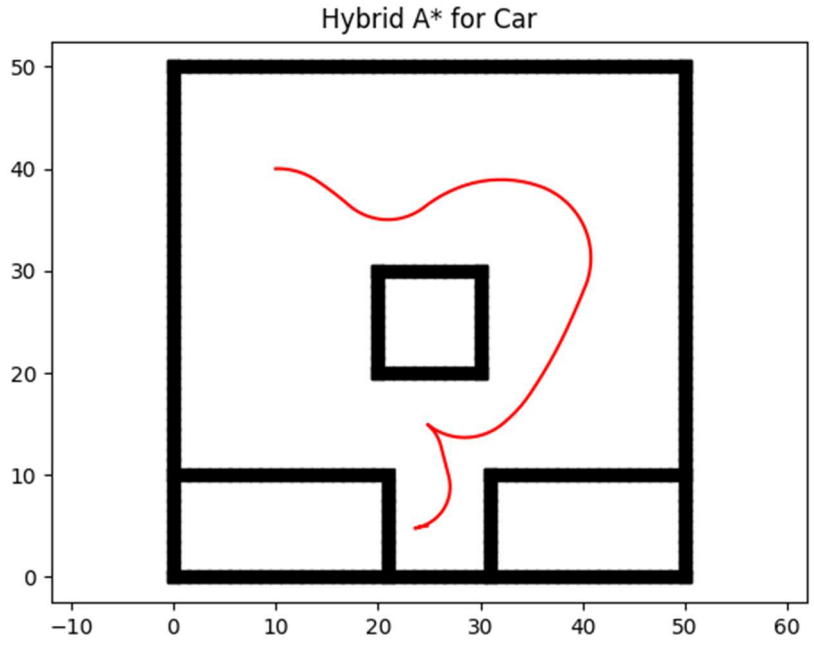
For our next experiment we use a car with Ackerman steering kinematics. This also traverses the same terrain but is challenging for the car to parallel park due to its non-skid kinematic model and furthermore its dimensions. The path followed by our car can be seen below. Kinematic equations used:

The position (x, y) and yaw ϕ of the test subject:

$$\dot{x} = v\cos\phi$$
$$\dot{y} = v\sin\phi$$
$$\dot{\phi} = v(\tan\psi)/l$$

where l is the wheelbase and ψ is steering angle.
The constraint equation is $\dot{x}\sin\phi - \dot{y}\cos\phi = 0$.





The above two images show the traversed path. As we can see the chosen path of the car is way wider than the di-wheel robot which is due to the steering and dimension constraints it has.

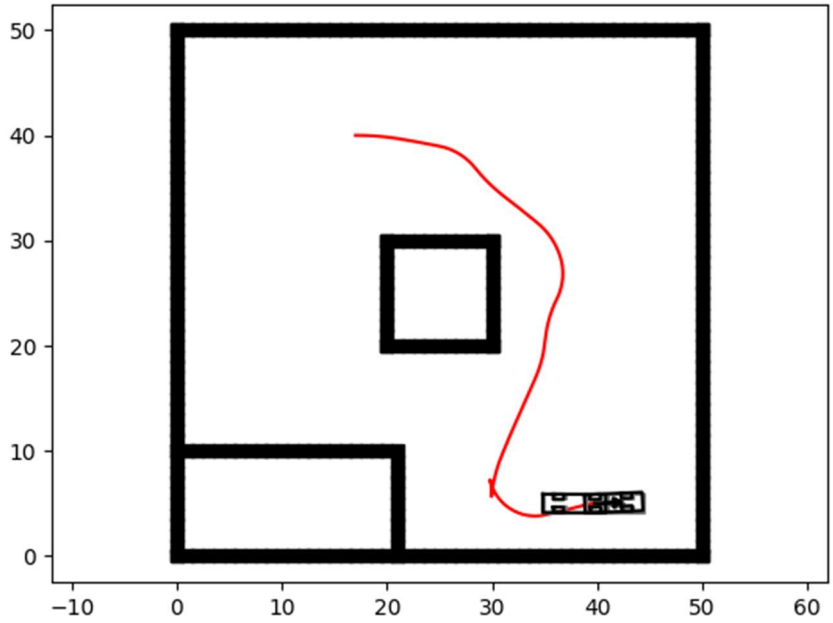
Truck with Trailer

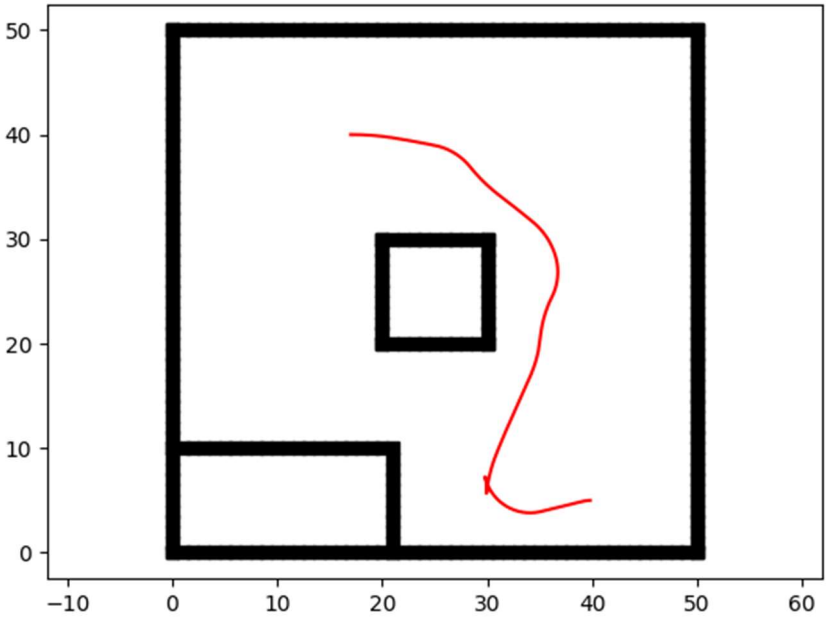
Finally, the trailer. We implemented the trailer kinematics for the final test subject in our experiment. The figures shown below depict the path traversed. For this experiment we remove the bottom right vehicle from our map. The kinematic equations used are:

The position (x, y) and yaw ϕ_1 (truck), ϕ_2 (trailer) of the test subject are:

$$\dot{x} = v\cos\phi_1$$
$$\dot{y} = v\sin\phi_1$$
$$\dot{\phi_1} = v(\tan\psi)/l_1$$
$$\dot{\phi_2} = -v(\sin(\phi_2 - \phi_1))/l_2$$

where l_1 is the wheelbase and l_2 is distance between rear axle and trailer axle and ψ is steering angle.
The constraint equation is $\dot{x}\sin\phi_1 - \dot{y}\cos\phi_1 = 0$.



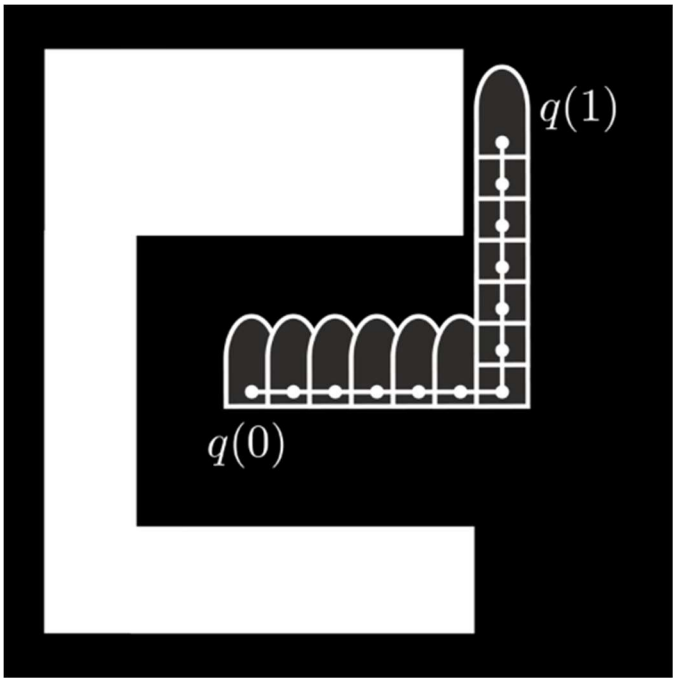


The observable discontinuity at the end of the drawn path is not actually a “discontinuity”. The reeds-shepp curves used reversed our test subject at that point i.e. negative velocity to park the trailer at the desired location and orientation.

Appendix

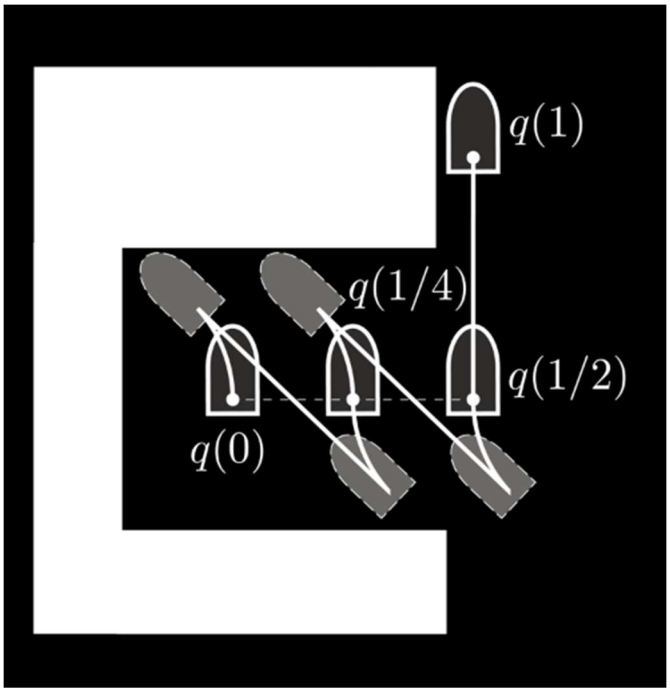
Reeds-Shepp Curves

Reeds-Sheep Curves are used to obtain the shortest path in an obstacle-free setting. Consider the below example:



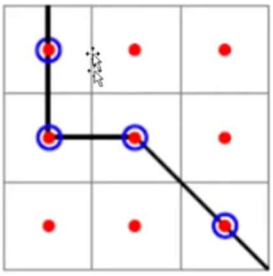
Here we have to go from q(0) to q(1) which is planned out by a simple planner in this case. We are using Hybrid A* for planning in our algorithm. Now as we can see the position from q(0) to q(1/2) is constrained by the outer walls. We divide this into smaller sub-goals to reach q(1/2) and then traverse on our previous planned path from q(1/2) to q(1) as shown below.

We could have used Dubin Curves instead of Reeds-Shepp which is basically Reeds-Shepp excluding the reverse motion i.e. negative velocity for our test subject, but having reverse motion available to our subjects gave us much more optimised paths.

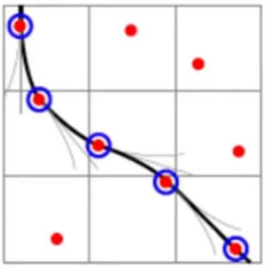


Hybrid A*

We use Hybrid A star to plan out the path for our test subjects as it takes into consideration the various constraints which our test subject might have for example maximum steering angle etc. Hence rather than generating the next node which would be a sharp and discrete turn as done by A-star, we get a much smoother and continuous one. Though we don't use the discrete points in choosing the next node, we still keep a queue of the discrete points as it helps us to check the already visited nodes. The visited node can differ by a few decimal values when compared with the continuous queue but the discrete queue helps us skip reiteration over the already visited nodes. A generic difference between A* and Hybrid A* can be seen below:

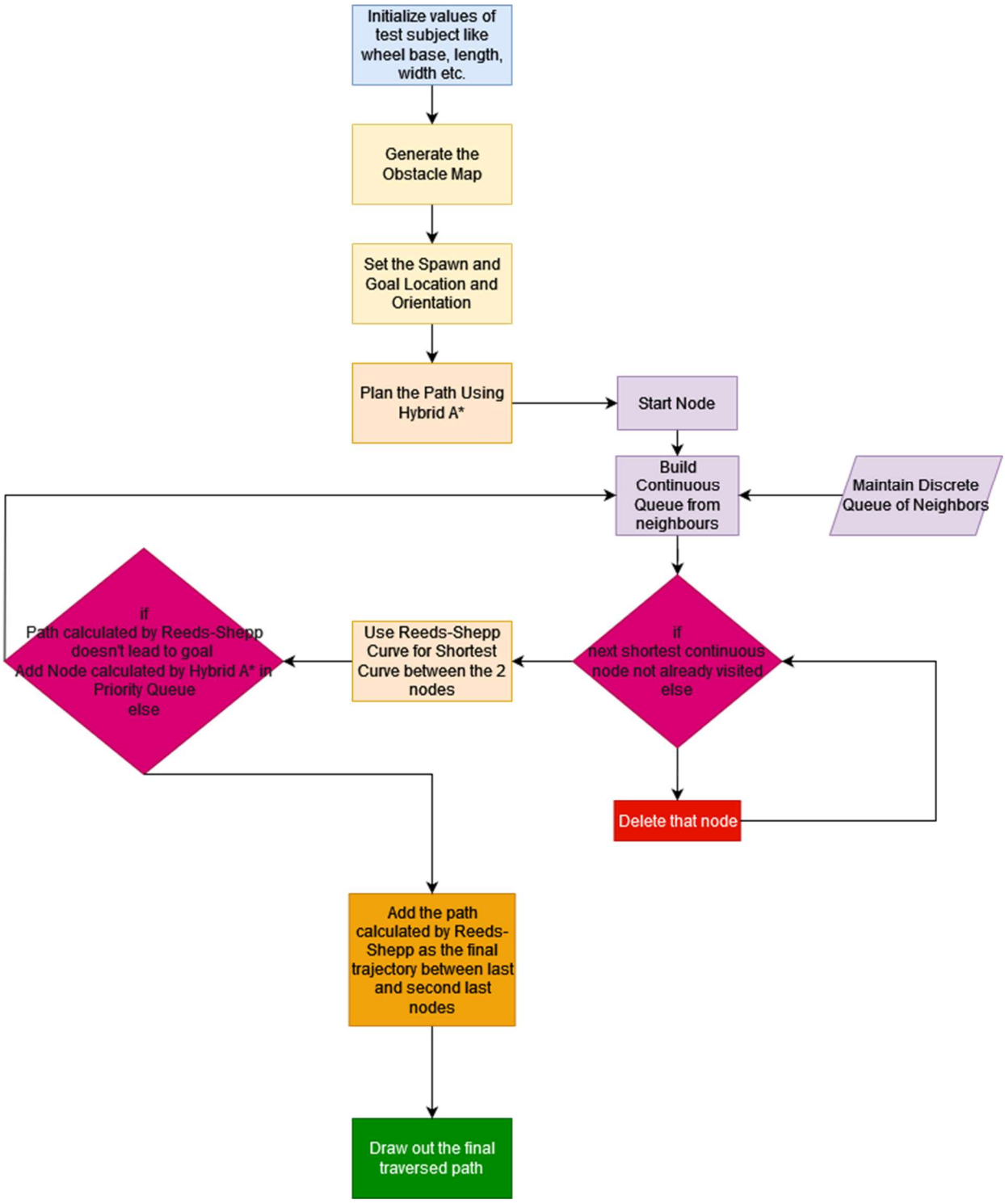


A*



Hybrid A*

Flow Diagram



References

[1] https://ai.stanford.edu/~ddolgov/papers/dolgov_gpp_stair08.pdf
[2] "*Optimal paths for a car that goes both forwards and backwards*",James Alexander Reeds, III and Lawrence A. Shepp, PACIFIC JOURNAL OF MATHEMATICS,Vol. 145, No. 2, 1990.
[3] http://planning.cs.uiuc.edu/node822.html
[4] zhm-real implementation of reeds-shepp and hybrid a-star.