Contents

1 Introduction

A wheeled mobile robot is a type of robot that is designed to move around on wheels or tracks. These robots can be used for a wide variety of tasks, such as transportation, inspection, surveillance, and many others. Wheel-based robots are a common choice for mobile robotics because they are relatively simple to design and build, and they can handle a variety of terrain types. They are also generally easier to control than other types of mobile robots, such as legged robots or flying robots. There are many different types of wheeled mobile robots, ranging from small, lightweight robots that can be carried by a single person to large, heavy-duty robots that can carry heavy payloads. The number and configuration of the wheels can vary, with some robots using two wheels, four wheels, or more. The wheels can also be powered or non-powered, and they can be designed to move in different ways, such as by rolling, skidding, or sliding. In addition to the wheels, wheeled mobile robots typically also include sensors, actuators, and a control system to enable them to navigate and perform tasks. They may also include other equipment, such as cameras, laser scanners, or grippers, depending on their intended application.

Trajectory tracking, path following, and point stabilization control are all techniques that are used to control the motion of a wheeled mobile robot. Trajectory tracking involves controlling the motion of the robot to follow a specific path or trajectory. This can be useful for tasks such as following a predetermined route, navigating to a specific location, or executing a predefined motion. Trajectory tracking typically involves designing a control law to adjust the robot's velocity and orientation based on the desired trajectory and the robot's current position and orientation. Path following is similar to trajectory tracking, but it involves following a path rather than a specific trajectory. A path is typically defined as a series of waypoints or markers that the robot should follow. Path following can be used for tasks such as navigating through a cluttered or complex environment or following a path that is not known in advance. Point stabilization control involves controlling the robot's motion to keep it stationary or hovering at a specific point. This can be useful for tasks such as maintaining a stationary position while performing other tasks or holding a specific location while waiting for further instructions. Point stabilization control typically involves designing a control law to adjust the robot's velocity and orientation based on its current position and orientation, as well as any external forces that might cause it to drift or move. These techniques can be implemented using a variety of control methods, such as feedback control, model-based control, or hybrid control. The specific method used will depend on the characteristics of the robot and the requirements of the task. Some of the well-known controllers for a car like mobile robots are the Lyapunov-based controller, model predictive controllers, sliding mode controllers, feedback linearization controllers, Geometry based controllers like Stanley controllers and pure pursuit controllers and others.

In this work, a path-following controller for a car-like wheeled mobile robot has been designed as a motion controller. In this process, first, a Lyapunov-based controller has been designed and global stability has been proved using La Salle’s Principle. Then, to minimize the errors, optimization has been undertaken in order to tune the control gains. Finally, to aid the controller the motion planner, in this case, a path planner has been articulated so that the controller can follow an obstacle-free path.

2 Problem Formulation

In this application, a bicycle kinematic model has been employed to where the controller will work on. The bicycle model represents the robot's motion as a combination of translational and rotational motion, similar to the motion of a car. In this model, the robot resembles a bicycle with front and rear wheels. The position and orientation of the robot can be described using a body frame that is attached to the centre of rear axle. So, according to figure (1), The position of the robot can be represented by the rear wheel coordinates (x, y) with respect to the world frame (X, Y), and the orientation can be represented by the angle θ. The coordinate of the front wheel is (x_f, y_f) and l shows the distance between the front and rear axle points. The robot's velocity can be represented by the linear velocity v and the steering angle is represented by δ. The non-holonomic constraints for this problem can be given by (1),

$$\dot{x}\sin(\theta + \delta) - \dot{y}\cos(\theta + \delta) - \dot{\theta} l \cos\delta = 0 \tag{1}$$

The solution of this equation thus leads to the state equation for this problem which is given by,

$$\begin{aligned} \dot{x} &= v\cos\theta \\ \dot{y} &= v\sin\theta \\ \dot{\theta} &= \frac{v}{l}\tan\delta \end{aligned} \tag{2}$$

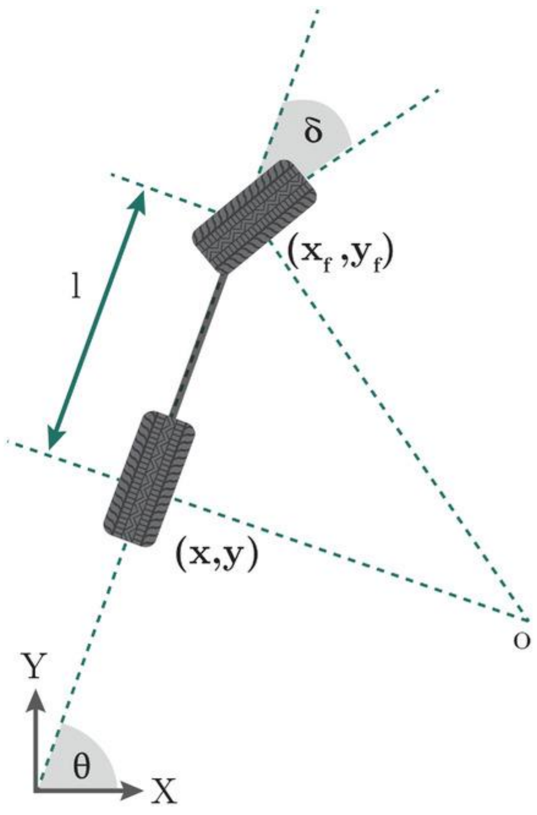

Figure 1: Bicycle model of car like mobile robot [1]

From the equation (2), it is evident that, $\delta = \frac{\pi}{2}$ is a singular point. Therefore, for this problem, $|\delta_{max}| < \pi/2$. The control input can be written as

$$u = \{(v, \delta): u_{min} < u < u_{max}\} \qquad (3).$$

If there is a path $\rho\ (x_d, y_d, \theta_d)$ path following control problem can be defined as,

Given there are states X given by according to (2) and control input according to (3) and a reference path as $\rho\ (x_d, y_d, \theta_d): R \rightarrow R^n$. The control objective is to design u in such a way that, $\lim_{t\to\infty} \|X - \rho\| = 0$. In addition, it is required to ensure that, the system will be stable for every point in the state space.

3 Controller Design

Given the state equation and reference path, the error configuration of a car like mobile robot can be given by,

$$x_e = (x_d - x)\cos\theta + (y_d - y)\sin\theta \qquad (4)$$

$$y_e = -(x_d - x)\sin\theta + (y_d - y)\cos\theta \qquad (5)$$

$$\theta_e = \theta_d - \theta \qquad (6)$$

The error state equation can be found as follows,

$$\begin{aligned} \dot{x}_e &= (\frac{y_e}{L}\tan\varphi - 1)v + v_d\cos\theta_e \\ \dot{y}_e &= -x_e\left(\frac{v}{L}\right) tan\varphi + v_d \sin\theta_e \\ \dot{\theta}_e &= \dot{\theta}_d - \frac{v}{L}\tan\varphi \end{aligned} \qquad (7)$$

As mentioned earlier, the control objective of the problem is to keep the errors to zero. So, the objective will be fulfilled if the equilibrium point X^* ends up to zero. Hence the equilibrium point is given by,

$$\lim_{t\to\infty} X^* = \lim_{t\to\infty} \|X - \rho\| = (x_e, y_e, \theta_e) = (0,0,0) \qquad (8)$$

Let a Lyapunov candidate be,

$$V = \frac{1}{2}x_e^2 + \frac{1}{2}y_e^2 + (1 - \cos\theta_e) \tag{9}$$

Then,

$$\dot{V} = \dot{x}_e x_e + \dot{y}_e y_e + \dot{\theta}_e \sin\theta_e \tag{10}$$

Theorem 1. Consider the error state equation be given by the equation (7) and reference path be ρ. Let the control law be as follows,

$$v = v_d \cos\theta_e + K_x\, x_e \tag{11}$$

$$\varphi = \tan^{-1}[\frac{L}{v_d \cos\theta_e + K_x\, x_e}\{\dot{\theta}_d + \frac{v_d}{K_y}(K_y y_e + K_\theta \sin\theta_e)\}] \qquad (12).$$

Where $K_x > 0, K_y, > 0\ and\ K_\theta > 0$ are control gains. Then the equilibrium point given by the equation (8) will be globally asymptotically stable.

Proof. From the inspection of equation (9), it is evident that, $V = 0$ at $X = X^*$ and $V = 0$ at $X = X^*$. Now putting the values of equation (7) in the equation (10) we get,

$$\dot{V} = -x_e v + x_e v_d \cos\theta_e + y_e v_d \sin\theta_e + \dot{\theta}_d \sin\theta_e - \left(\frac{v}{L}\right) tan\varphi \sin\theta_e \quad (13)$$

Putting the value of equation (11) and (12) in equation (13),

$$\dot{V} = -K_x x_e^2 - \frac{K_\theta}{K_y}(\sin\theta_e)^2 \qquad (14)$$

Now, we can see that, $\dot{V}$ is negative semi-definite for all values of X. So, the equilibrium point is said to be stable. Let, we define D in such a way that,

$$D = \{\begin{bmatrix} x_e \\ y_e \\ \theta_e \end{bmatrix} : x_e = 0\ and\ \theta_e = 0\}$$

And let, M is the largest invariance set to D. If we start in D, then, $x_e = 0\ and\ \theta_e = 0$. Putting these values in equation (9)-(11) we get,

$$\dot{x}_e = 0, \dot{y}_e = 0\ and\ \dot{\theta}_e = 0$$

From equation (11) we can write, $\dot{\theta}_d = \frac{v}{L}\tan\varphi$

Or, $\dot{\theta}_d = \dot{\theta}_d + \frac{v_d}{K_y}\left(K_y y_e + K_\theta \sin\theta_e\right)$ [Putting the value of v and φ]

Or, $v_d\, y_e + v_d \frac{K_\theta}{K_y}\sin\theta_e = 0$

Or, $y_e = -\frac{K_\theta}{K_y}\sin\theta_e$

$y_e = 0$ [as θ_e = 0]

So, M = [0 0 0] which is equal to equilibrium point E = E*. Hence the equilibrium point is locally asymptotically stable. To prove that the equilibrium point is asymptotically stable, we have to show that V is radially unbounded. That is, when $\|X\| \rightarrow \infty, V \rightarrow \infty$ If, $\|X\| = \sqrt{x_e^2 + y_e^2 + \theta_e^2}$ goes to ∞, from observation it is seen that, $V = \frac{1}{2}x_e^2 + \frac{1}{2}y_e^2 + (1 - \cos\theta_e)$ also goes to ∞. Hence, we can conclude that, the equilibrium point is globally asymptotically stable. **(Proved)**

No, to enhance the result of the control action, it is necessary to conduct optimization. In this case, a performance criterion, specifically, root mean square error has been minimized to obtain such control gains so that errors could be closer to zero. The optimization problem can be formulated as follows,

$$\min J\left(K_x, K_y, K_{theta}\right) = \int_{\rho(x_0,y_0,\theta_0)}^{\rho((x_f,y_f,\theta_f)} \sqrt{(x_d - x)^2 + (y_d - y)^2 + (\theta_d - \theta)^2}$$

$$S.T, \qquad K_x > 0, K_y > 0, K_{theta} > 0$$

4 Result

For path planning, a hybrid method of A^* and polynomial approaches have been employed. However, the Dijkstra algorithm has been simulated which produce computationally expensive result than the hybrid method. In fig. 2, the black, white, red, blue, red, grey, and green colors represent the obstacles, free space, search space, space on which the search has been executed, path and starting point respectively. The Dijkstra algorithm searches all the points of its search space which results in a more expensive algorithm. On the other hand, Astar does not explore all the space rather takes some wise guess to search. Reasonably, the path generated by the two algorithms is not identical.

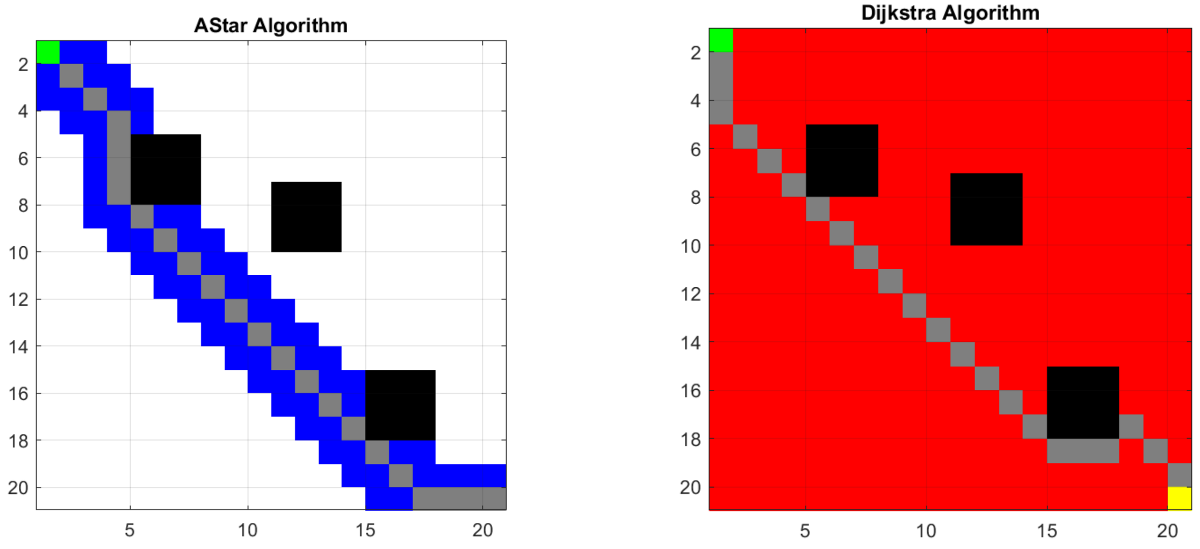

Fig. 2: Simulation of A^* and Dijkstra Algorithm

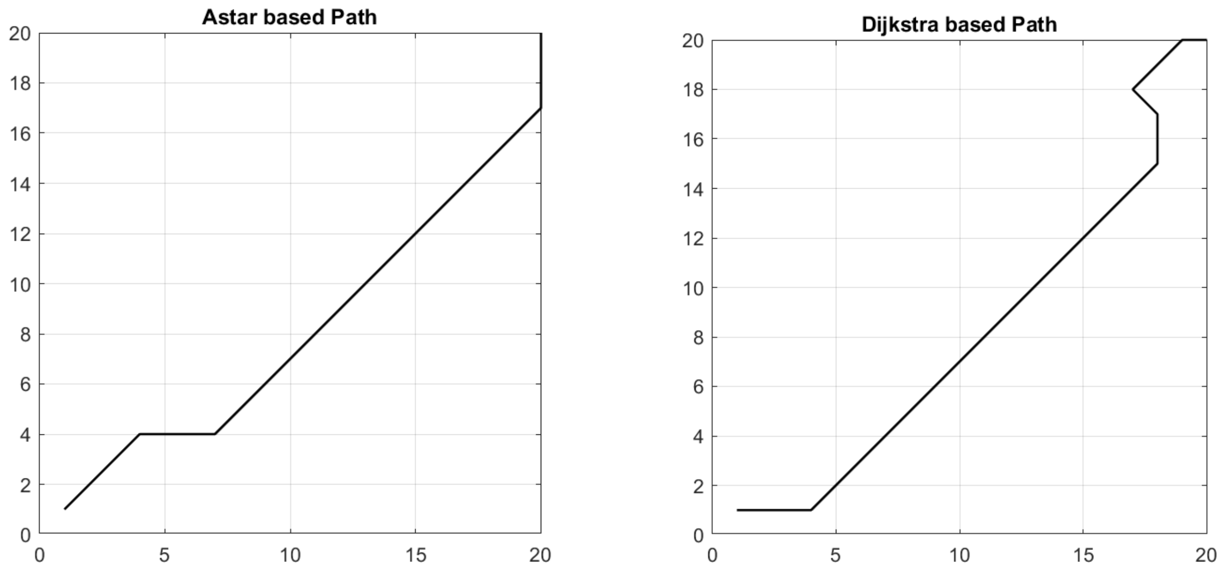

Fig. 3: The resultant path from the A^* and Dijkstra Algorithm

Surely, these paths do not give any information about the orientation which necessitates a polynomial which could represent the path with the conformation to steering and nonholonomic constraints. Furthermore, at the time of control action, as the path is not smoother, the result will be discontinuous. Therefore, a polynomial-based path planner has been adopted in this case. The results of the controller are given below.

Table 1: Simulation parameters and the result of optimization

Parameter Name	**Values**
l	2 meters
$(x_{start}, y_{start}, \theta_{start})$	(0,0,0)
$(x_{goal}, y_{goal}, \theta_{goal})$	(20.20,0)
K_x^*	0.68
K_y^*	0.22
K_θ^*	2.6
$RMSE^*$	14.43

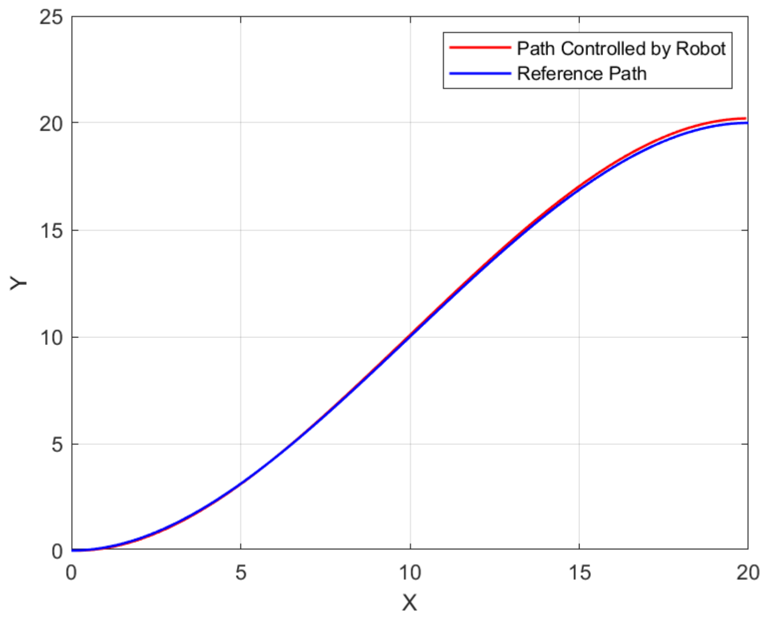

Fig. 4: Path following action

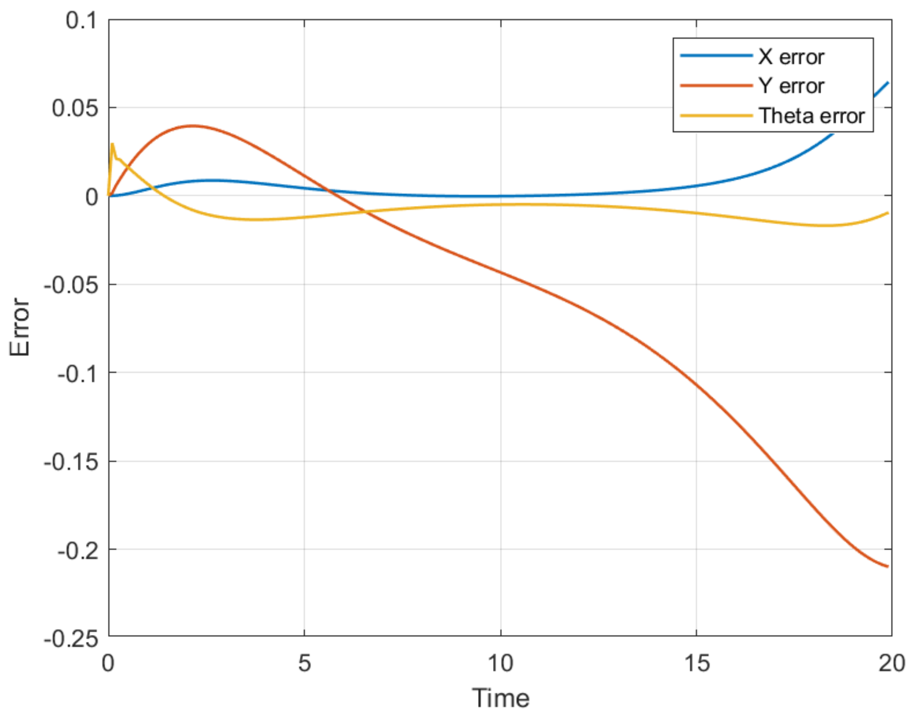

Fig. 5: Error Vs Time (Second)

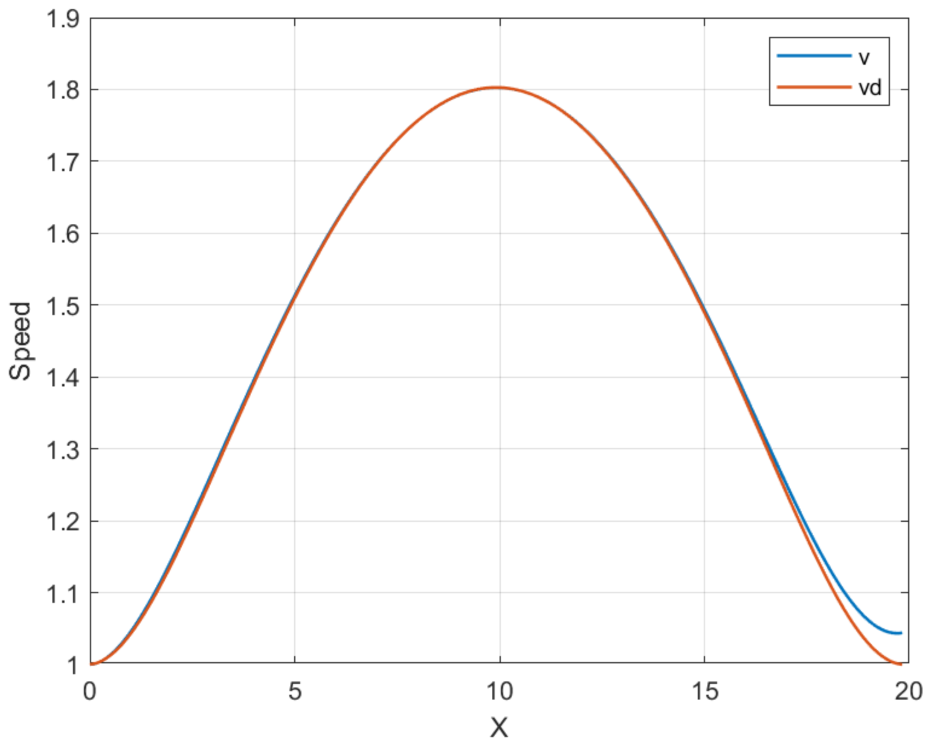

Fig. 6: Speed (m/s) Vs X(m)

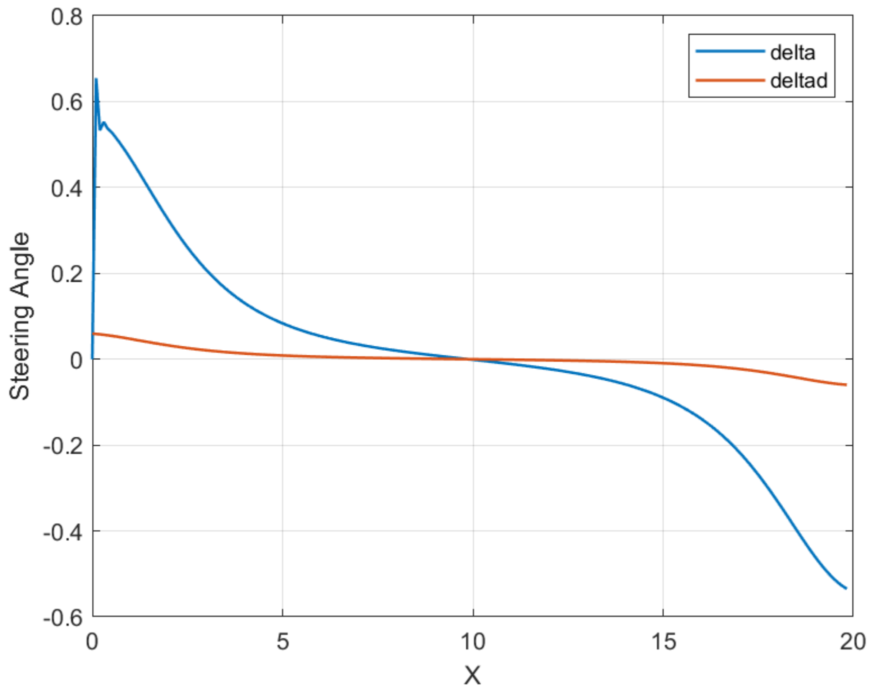

Fig. 7: Steering Angle(rad) Vs X(m)

In Fig. 5, we can see that x and θ errors are limited to some values, however, y errors are subject to further improvement. From Fig 7, the desired steering can not be fully achieved by the control.

Source Code

AStar Path Planning

```
clc
close all
clear
tic
%% map and obstacle
int_map = false(20);
int_map(5:7,5:7) = true;
int_map(15:17,15:17) = true;
int_map(7:9,11:13) = true;
%% start and goal pos and color
start_coords = [1 1];
dest_coords = [20, 20];
dx = 1;
dy = 1;
cmap = [1 1 1; ...
    0 0 0; ...
    1 0 0; ...
    0 0 1; ...
    0 1 0; ...
    1 1 0; ...
    0.5 0.5 0.5];
```