HOMEWORK-3

Name : Mostroor Mofiz Arman

ID : 1921079642

Course : CSE495A

Section : 1

Submitted to :

Dr. Shahnewaz Siddique (SnS1)
Associate Professor
Department of Electrical & Computer Engineering
North South University

Date : 22-10-2024

2

Ans to the QNO - 1

(a)

A* method requires more computational resources because it explores a grid or graph by calculating and storing cost values ($g(n)$, $h(n)$ and $f(n)$) for every node. This can become computationally expensive for high dimensional spaces or fine grids. On the other hand, differential flatness uses mathematical transformations to directly generate trajectories based on a flat output. This avoids iterative exploration, making it computationally efficient compared to A*. Therefore, A* method demands more computational resources due to its graph-based exploration and memory usage.

(b)

A* method is better for obstacle avoidance. A* explicitly evalutes potential paths, ensuring they avoid obstacles. It systematically searches for the shortest path while taking obstacles into account. On the other hand, differential flatness doesnot inherently account for obstacles unless combined with an ~~ext~~ external planing method. Therefore, A* method is better for obstacle avoidance.

(c)

Differential flatness is better for movement in obstacle free 3d space. In an obstacle-free 3D space, differential flatness is highly efficient, generating smooth and dynamically feasible trajectories without iterative search. On the other hand, A* method is computationally inefficient in open, obstacle-free space due to unnecessary exploration of nodes. Trajectory generated may not be inherently smooth and may require post-processing. For those drawbacks, we didn't chose A* method for movement in obstacle free 3d space. Therefore, differential flatness is better for movement in obstacle free 3d space.

Ans to the QNO-2

Given,

Lyapunov function: $V(\rho, \alpha, \delta) = \frac{1}{2}\rho^2 + \frac{1}{2}(\alpha^2 + K_3\delta^2)$

Time derivative, $\dot{V} = \rho\dot{\rho} + \alpha\dot{\alpha} + K_3\delta\dot{\delta}$

From 3.9 equation,

$$\dot{\rho} = -v\cos\alpha$$

$$\dot{\alpha} = \frac{v\sin\alpha}{\rho} - \omega$$

$$\dot{\delta} = \frac{v\sin\alpha}{\rho}$$

From 3.11 equation the control inputs are

$$v = K_1\rho\cos\alpha$$

$$\omega = K_2\alpha + K_1\frac{\sin\alpha\cos\alpha}{\alpha}(\alpha + K_2\delta)$$

Now,

$$\rho\dot{\rho} = \rho(-v\cos\alpha) = -\rho(K_1\rho\cos\alpha)\cos\alpha = -K_1\rho^2\cos^2\alpha$$

$$\alpha\dot{\alpha} = \alpha\left(\frac{v\sin\alpha}{\rho} - \omega\right)$$

substitute v and ω,

$$\alpha\dot{\alpha} = \alpha\left(\frac{K_1\rho\cos\alpha\sin\alpha}{\rho} - \left(K_2\alpha + K_1\frac{\sin\alpha\cos\alpha}{\alpha}(\alpha + K_3\delta)\right)\right)$$

$$= \alpha\left(K_1\cos\alpha\sin\alpha - K_2\alpha - K_1\sin\alpha\cos\alpha - K_1K_3\delta\frac{\sin\alpha\cos\alpha}{\alpha}\right)$$

$$= -K_2\alpha^2 - K_1K_3\delta\sin\alpha\cos\alpha$$

$$\therefore K_3\delta\dot{\delta} = K_3\delta\left(\frac{v\sin\alpha}{\rho}\right) = K_3\delta\frac{K_1\rho\cos\alpha\sin\alpha}{\rho} = K_3K_1\delta\sin\alpha\cos\alpha$$

So,

$$\dot{V} = -K_1\rho^2\cos^2\alpha - K_2\alpha^2 - K_1K_3\delta\sin\alpha\cos\alpha + K_3K_1\delta\sin\alpha\cos\alpha$$

$$= -K_1\rho^2\cos^2\alpha - K_2\alpha^2$$

Since $K_1 > 0$ and $K_2 > 0$, $-K_1\rho^2\cos^2\alpha$ and $-K_2\alpha^2$ are negative or zero. Therefore $\dot{V} \leq 0$ and it's strictly negative unless $\rho = 0$ and $\alpha = 0$. Thus the system converges to the origin.

Ans to the QNO-3

Given,

$$\dot{x}_1 = -x_1 + x_2^3$$

$$\dot{x}_2 = -x_2 + u$$

Lyapunov function, $V = \frac{1}{2}x_1^2 + \frac{1}{4}x_2^4$

Time derivative of V, $\dot{V} = \frac{\partial V}{\partial x_1}\dot{x}_1 + \frac{\partial V}{\partial x_2}\dot{x}_2$

$\therefore \frac{\partial V}{\partial x_1} = x_1$ and $\frac{\partial V}{\partial x_2} = x_2^3$

Now, substituting $\dot{x}_1$ and $\dot{x}_2$,

$$\dot{V} = x_1(-x_1 + x_2^3) + x_2^3(-x_2 + u)$$

$$\Rightarrow \dot{V} = -x_1^2 + x_1x_2^3 - x_2^4 + x_2^3 u$$

$\dot{V} < 0$ for all $(x_1, x_2) \neq (0,0)$,

$u = -x_2^3 - Kx_2$; where $K > 0$ is a positive gain.

Substituting u into $\dot{V}$,

$$\dot{V} = -x_1^2 + x_1x_2^3 - x_2^4 + x_2^3(-x_2^3 - Kx_2)$$

$$= -x_1^2 - x_2^4 - Kx_2^4$$

Since all terms in $\dot{V}$ are negative or zero.
Therefore, $\dot{V} \leq 0$ ensuring stability.

Thus, $u = -x_2^3 - kx_2$; where $k>0$ is a positive gain.

Ans to the QNO-4

A* grid-based search algorithm :

(1) Start from the initial node like start point and add it to open list where nodes to be evaluated.

(2) Assign a cost to the node using the formula

$f(n) = g(n) + h(n)$ where $g(n)$ is the cost from the start to the current node and $h(n)$ is the Heuristic estimate to the goal.

(3) Select the node with the smallest $f(n)$ from the open list and move it to the closed list which is already evaluated.

(4) Generate all possible neighbour nodes of the current node and if a neighbour is in the closed list, skip it.

(5) If not in the open list, calculate its $f(n)$ and add it to the open list.

(6) If already in the open list with a higher cost, update the cost.

(7) Repeat step (3) to step (6) until the goal node is reached or the open list is empty.

Cons of the A* method :

(1) Computationally expensive for large grids.
(2) Performance heavily depends on the quality of the heuristic.
(3) Can become memory-intensive as the open and closed lists grow.

Ans to the QNO - 5

Probabilistic Road Map (PRM) Sampling-based motion planning method :

(1) Randomly sample points in the configuration space to generate collision-free nodes.
(2) Connect sample nodes to their nearest neighbours using local planners and check for collisions.
(3) From a graph where nodes represent sampled points and edges represent feasible connections.
(4) Use graph search to find the shortest path from the start to the goal node.

9

Cons of the method:

(1) Requires a preprocessing phase to build the road-map, which is computationally expensive.

(2) Not efficient in dynamic environments where obstacles or goals may change.

(3) Quality of the solution depends on the density and distribution of sampled points.

Ans to the QNO-6

Rapidly-exploring Random Tree (RRT) sampling-based motion planning method:

(1) Start with a tree rooted at the initial position.

(2) Randomly sample a point in the configuration space.

(3) Find the closest node in the tree to the sampled point.

(4) Extend the tree by moving from the closest node towards the sampled point by a small

steps, ensuring collision-free paths.

(5) Repeat step (2) to step (4) until the tree reaches the goal region or the maximum number of iterations is reached.

Cons of the method:

(1) Paths are often suboptimal and require smoothing or post-processing.

(2) Poor coverage in narrow or constrained spaces.

(3) Can take a long time to coverage if the random sampling is inefficient.