Mid Term Answer

1. For this vehicle, the motion will be easier to plane in the workspace. Thus, the topology becomes an R1S2, that is controlling only the two angles of the agent and r the distance from the start point. The system is them reduced to a 3d coordinate (polar coordinate).

Explanation: The aerial vehicle shows in this problem as 3 coordinates (x, y, z) and 2 rotational angles. Thus, the workspace of the will have a topology od R3S2. It would be difficult and required a lot of computer resources to plane the motion this as we have about 5 parameters to tweak.

However, from the equations given (dynamics of the system), the controller inputs, the constrains, with careful examination; one can see that the system can be controlled using the two angles of the rotation and a radius. In fact, the input to the system is the rate of change of the angles and the acceleration. One can reverse calculate the coordinate of the agent at every sampled angle and check if it is obstacle free.

From this, the workspace will be spherical ball will radius the distance from the reference point to the goal. Every sample point will be a subset of the spherical ball. The motion will be planned from a smaller subset spherical ball starting at the start point to the toward goal.

Radius

angle

obstacle

Figure: View of the 2d representation of the polar planning using RRT in R1S1 topology.



|  |  |  |
| --- | --- | --- |
| Planning Algorithms | Advantages | Disadvantages |
| Gradient descent planner with a potential function | * Can find the path to goal if no local minimum and one global minimum * Fast * No path smoothing needed | * Local minimum * Invalid points: the next point could be invalid due to constraints set. * Not easy to control controller input. |
| Wave-front planner | * Guaranteed to find path if it exists * Easy to implement * Can be implemented in workspace | * Little room to manipulate controller input because points are pre-sampled * Will need path smoothing |
| Probabilistic roadmap planner | * find path if it exists * Easy to implement * Can be implemented in the chosen workspace | * Can become computationally intensive * Is probabilistic complete * Could lead to many invalid samples thus not path due to constraint if sample points too sparse * Will need path smoothing * Not easy to control controller input. |
| Randomized tree-based planner | * Ability to control or tweak controller inputs * Could be fast depending on the version chosen * Will find path if it exists (this is time dependent) * Might not need path smoothing small step is chosen | * Could take too long to complete depending on the planning flavor. * Can be computationally intensive. * Probabilistic complete |

1. Pseudocode RRT\_Goal\_Bias

Text

Description automatically generated

1. Using the chosen algorithm, RRT goal bias, the planner should find the path to goal if it existed given enough time. This is guaranteed because we will sample some data points near goal at x% of time. Thus, if the algorithm is given enough time I should converge to goal.
2. With RRT Goal Bias, we sample a point in the space of the agent then we then choose a new P\_new sample that is in a radius r that we define along the closes point in our graph’s valid points. With this approach we can have control over the chosen P\_new. Thus, for the point p\_new, the team can optimize the controller input that allow to get the best point a.k.a best edge. The team can then construct the graph with points that are optimal. We can define each point with more than one optimal parameter.

For example, if each edge has the following structure

Struct edge {

tuple p1;

tuple p2;

float time;

float distance;

float other\_parm;

}

The team can then find best path using for example time or distance of each edge.

1. Dynamic of the drone in the environment:

* Before passing through rope plane:

Before passing through the rope plane, the dynamic of the drone should not be different. However, there would be a need to design a controller that control each element of the drone independently. That is, the pitch, roll and yaw would be controlled by independent controllers. This will allow adjusting input to the plant (drone) without affecting directly other dynamics of the drone.

The drone will also need to keep a high altitude with respect to the goal point altitude. In fact, if more than one motor is lost, the difficulty of flying the quadcopter should increase. The idea here is that if flying the quadcopter becomes (nearly) impossible, the system can calculate how much force is needed to truss itself as a projectile to the goal. Well, for this to work, one needs to flight above all known obstacle.

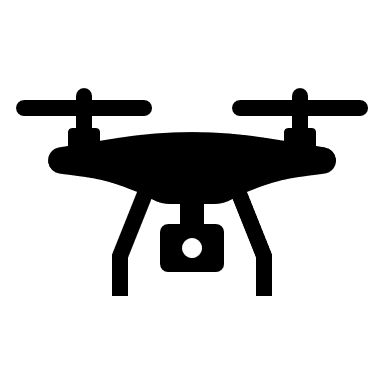
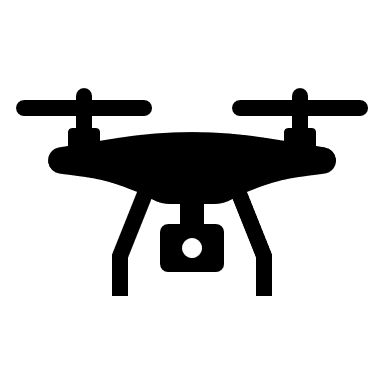
* After passing through rope plane:

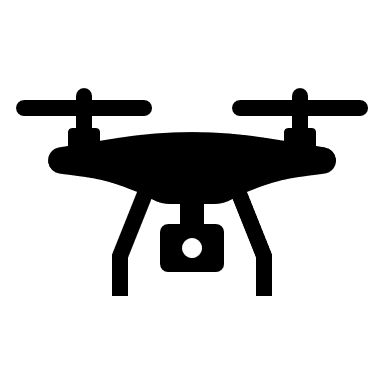
As hinted above, there are few options to consider after passing through the ropes plane. Before we get there, it is necessary to point out that the yaw control of the quadcopter has to be deactivated after impact. Thus, the quadcopter will be allowed to spin toward goal horizontally but freely. This could work fine with three propellers with angle out of the dynamics of the system.

With more than one propeller damaged, it would be almost impossible ( to not say simply impossible) to fight the quadcopter, however, while flying above all know obstacles, projectile motion can be used to achieve the goal point requirements.

When the quadcopter hit the rope, system need to calculate the pitch, roll position acceleration needed to simulate a free fall projectile motion using the remaining propeller to get to goal. Normal flying cannot be achieved at this stage.

Continue the flight





Projectile motion

Obstacle

Flight to higher altitude

rope

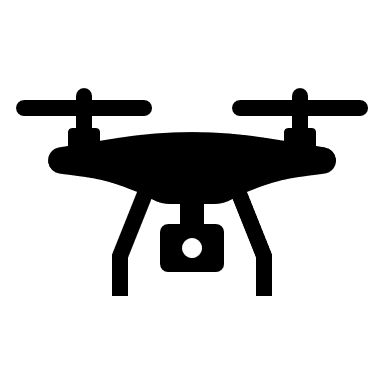


Figure: Representation of the flight scenario in 2d

1. *Pseudocode and description*

Description:

From algorithm in the first part, the following improvement can be made according to the dynamics of the quadcopter.

* First the algorithm needs to find a path to the goal with the higher altitude that give a clear view with no obstacle on the way.
* The quadcopter flight to that maximum altitude first from the start point
* During flight, at each step or controller input, the system needs to calculate the force needed to flight as a projectile (almost free fall) to the goal using the requirements.
* Always check for damage at each step
* In case of damage, record the reference plane and get a new path to the goal by considering the plan as an obstacle.
* When a motor is loss, flight as projectile (free fall but truss toward goal with remaining propellers) with horizontal spinning toward goal
* If 3 motors loss, there is no possibility to reach goal, free (vertical) fall.

Text

Description automatically generatedPseudocode:

Text

Description automatically generated

1. The planner in part g can find a path to goal if one exists but subject to how many motors are lost in the process. If 3 motors are lost, I will be impossible to flight. That is, as one cannot predict the location on the ropes, it is not possible to predict how many motors we could be lost getting to the goal. So, the algorithm completion is related to the flying ability of the quadcopter. Thus, no I cannot guarantee that we can get to goal is a path exit