Markov Decision Process Formulation

Rounak Singh

Department of Electrical and Computer Engineering 10/23/20

University of Missouri Columbia

1. **Test Scenario:** We are considering a scenario where a a drone is hovering over a designated area of Surveillance.

   Purpose: Live Video Streaming during disaster response.

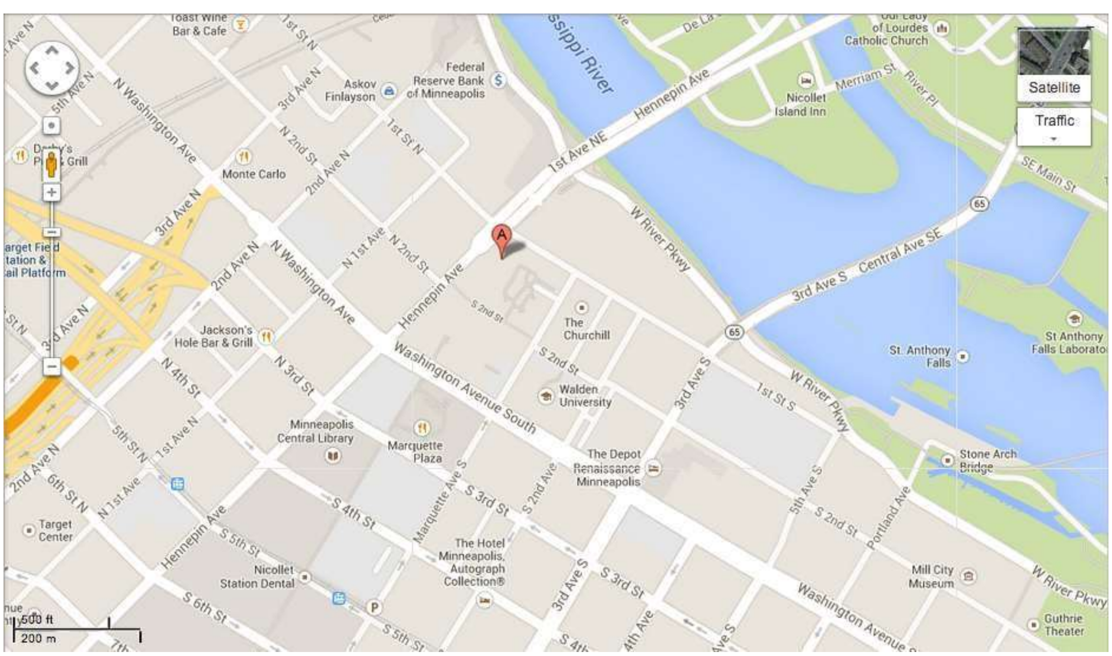
Goal: We aim to formulate a Finite Markov Decision Process for the Drone which starts from an initial state (random state) and then hovers over the designated area for some time and then lands down for *recharging.* (Terminal state).

In the Area which the Drone hovers, there are certain sub-areas within the larger area which are prone to vulnerabilities like signal-loss, cyber-attacks, high-altitude etc. which severely affect the drone's performance and make its functionality unreliable.

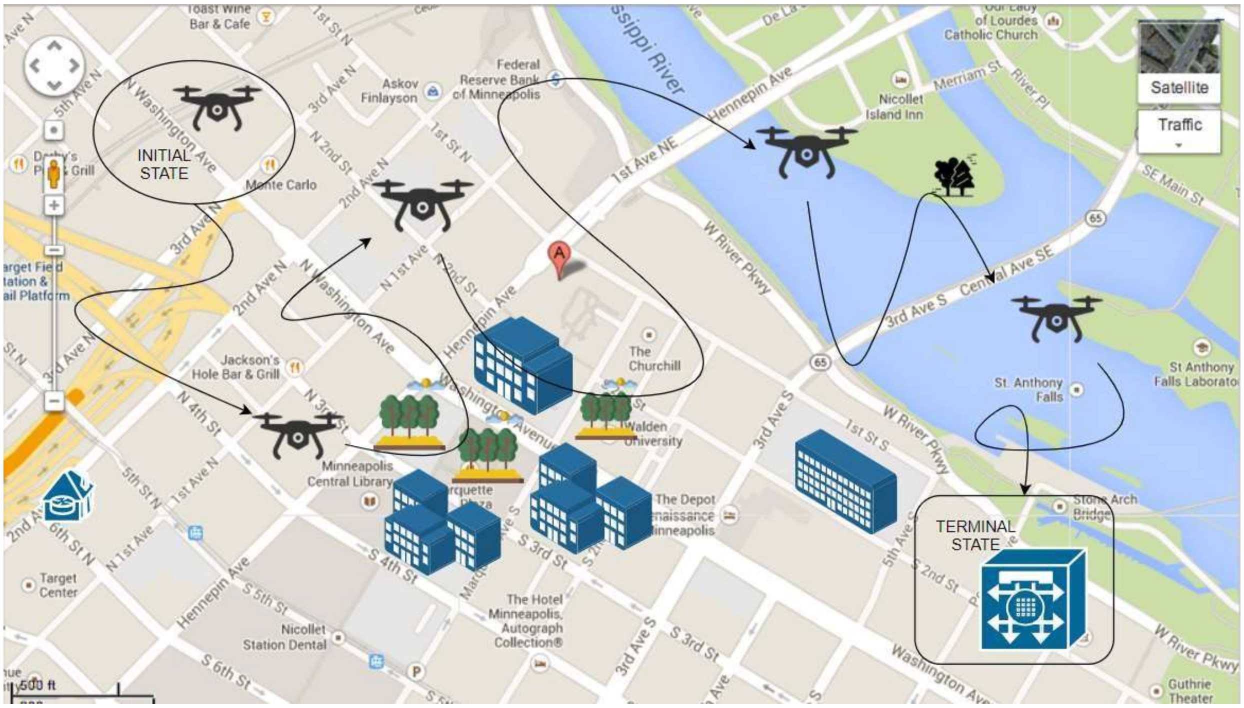
These sub-areas include clusters of building, trees, huge structures or obstacles of any kind which interrupt the drone's communication process with the Base station.

1.1 Problem Formulation:

Consider the following area is chosen for surveillance:



Drone Movement: Trajectory which it follows during surveillance



Precarious Area: Area prone to Cyber-attacks and Signal-Loss

2.Constituents:

M = {State, Action, Reward, Probability}

2.1 **Environment** = {States, Actions}

2.2 States:

Formulation: **s =** current state
s'= next state

Real-Time:

S0: Initial state

- Any random area (assumed to be S1)

S1: Hovering over Secure area

- High Battery
- Remain Stationary
- Perform Surveillance

S2: Hovering over Precarious area

- Immediately move to S1

S3: Terminal state

- Low Battery
- Stop Surveillance
- Recharge at Control Station

***Note: **S0 = S1** (Since, initial state is chosen randomly)

2.3 Actions:

Formulation: **a =** current state action
a'= next state action

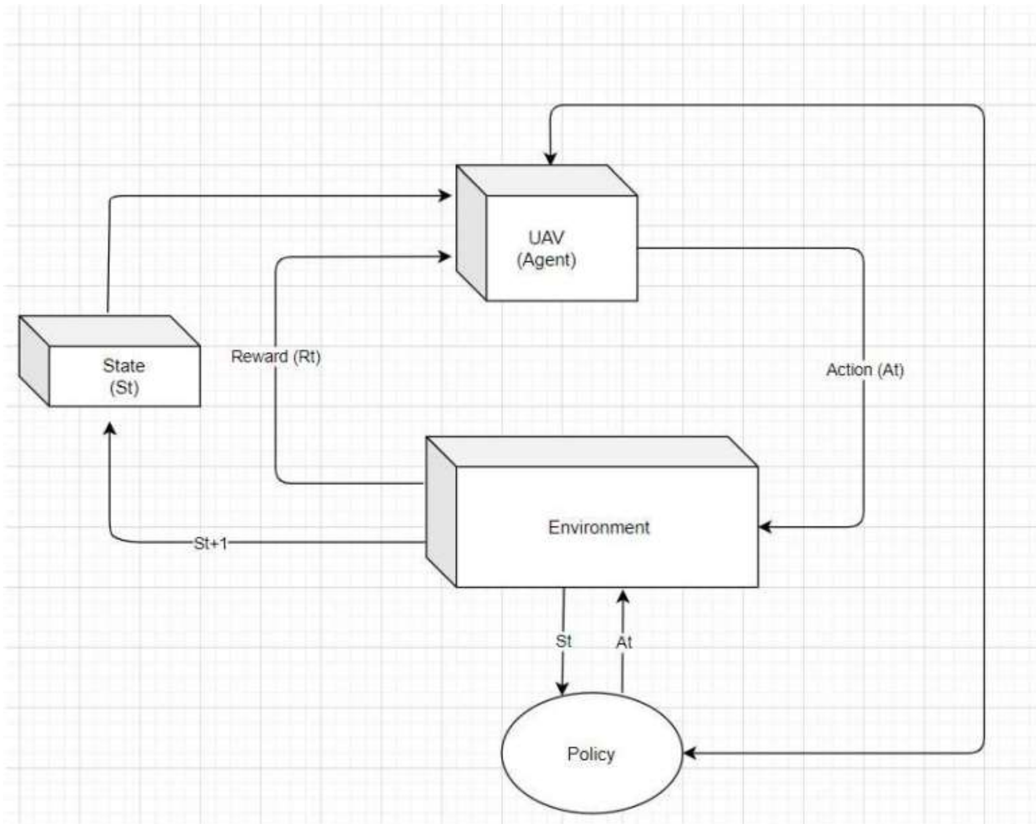
Real-Time:

A0: Remain Stationary
A1: Move
A2: Go to Terminal state

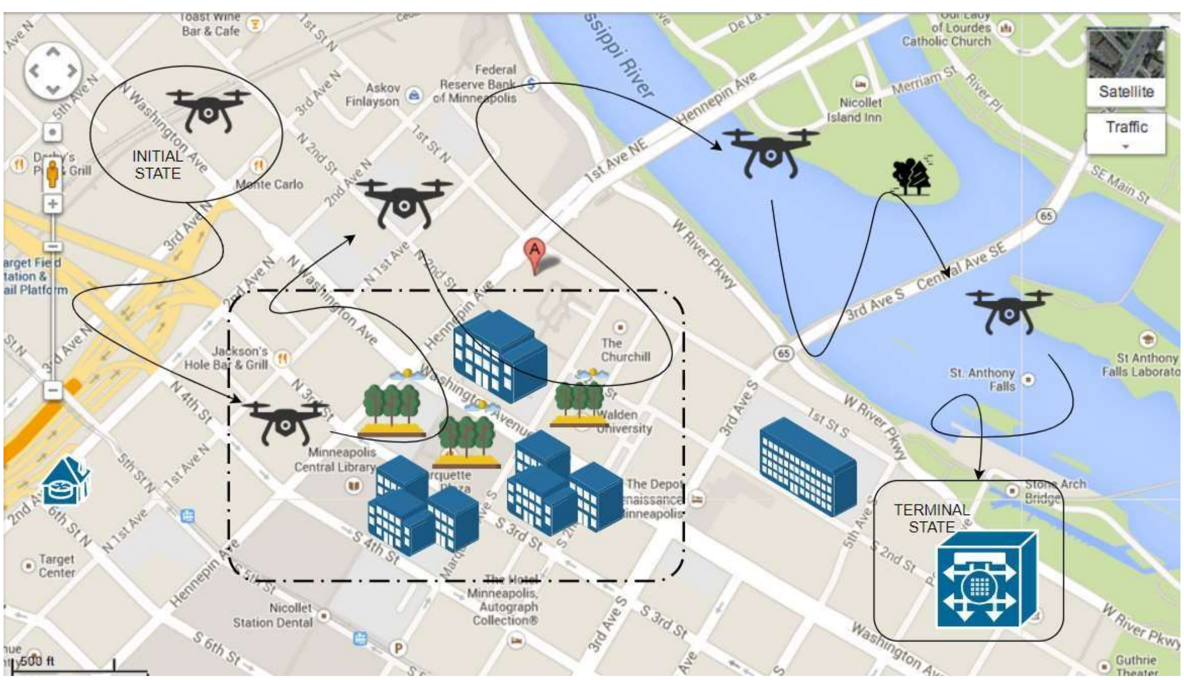
2.4 Rewards:

Rt(secure) = Reward obtained when entering secure region
Rt(network) = Reward obtained for maintaining a reliable network
Rt = Total reward

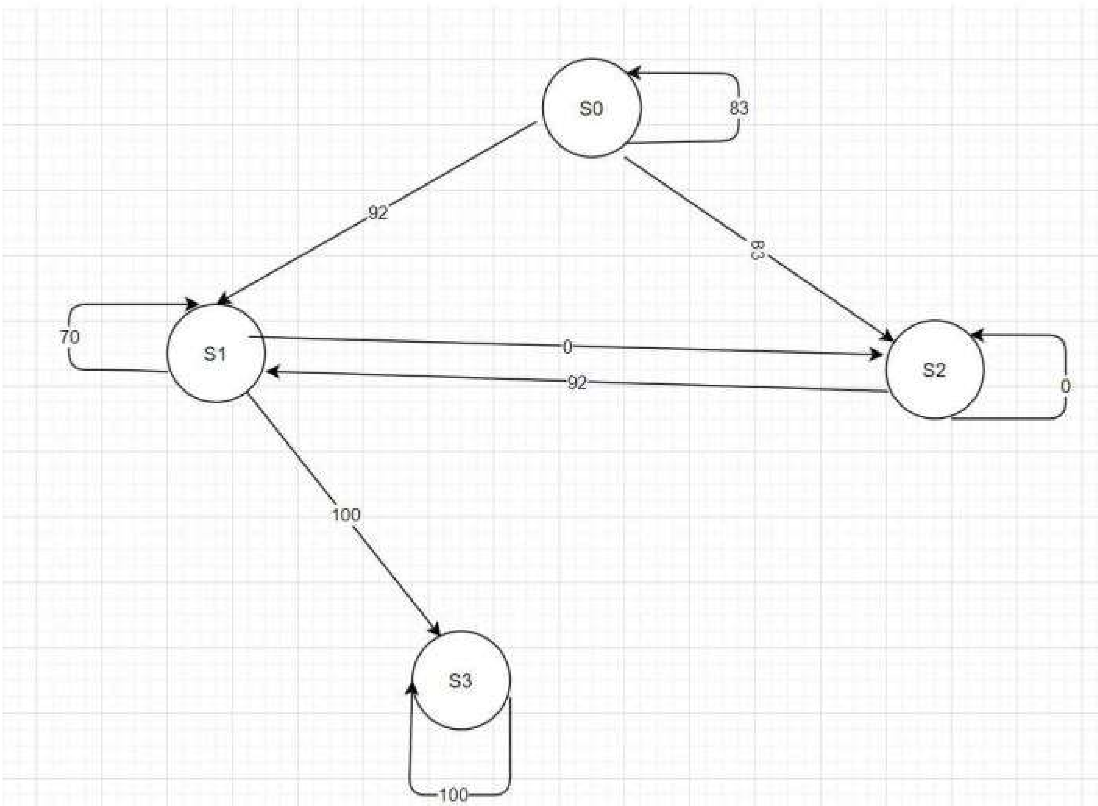
Block Diagram:



Overall Trajectory:



2.5 State Transition Graph :



2.6 Code: Reinforcement Learning using python

*****Refer to the Jupyter Notebook for Implementation*****