Project Report

Wind-Energy based Path Planning for Electric Unmanned Aerial Vehicles Using Markov Decision Processes

by Sushil Vemuri 19317

1. Motivation

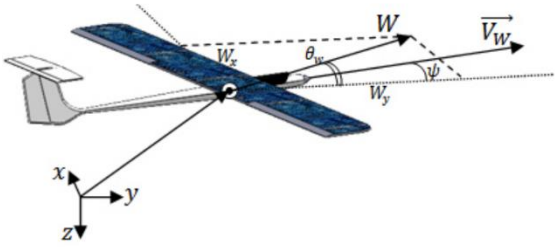
Unmanned Aerial Vehicles (UAVs) are small, electric-powered aircraft used in both military and civilian applications. Coastal or border surveillance, atmospheric and climate research, remote environment, forestry, agricultural, and oceanic monitoring and imagery for the media and real-estate industries are all possible applications for such aircraft. However, one of the most significant limitations of small UAVs is their flight endurance due to the limited amount of onboard (fuel/battery) that can be carried. For planning the best route, it is critical for these vehicles to harness fluctuating and unknown environmental circumstances (horizontal wind, vertical wind) in order to maximise flight length and minimise power usage. Due of the vehicle's small size, the unpredictable amplitude and direction of the wind can really cause uncontrollable forcing to be applied to it.

2. Introduction

The authors of this paper incorporate the uncertainty of the wind field into the wind model and plan using a Markov Decision Process (MDP). Because the wind velocity is unclear, the authors proposed that the next state be treated as a random variable, with a probability distribution generated over all adjacent cells. The motion planning problem is then to select the actions (horizontal and vertical actuation of the balloon) that minimise time to goal given these

transition probabilities from all states. The MDP finds the best immediate action for each current state in order to reduce the predicted cumulative time-to-goal.
Using a Gaussian model and a modified MDP technique, the suggested method provides the optimal power-based path, the path that uses the least amount of onboard energy while taking into account the fluctuation in wind magnitude and direction over time to reach a certain target point.
As a result, this problem is readily formulated as a Markov Decision Process (S; A; P; R), where S symbolises the set of potential states of the aircraft; A denotes the set of possible actions of the aircraft; The transition probabilities are represented by P. For each transition and action, R defines the expected immediate reward.

3. Problem Statement

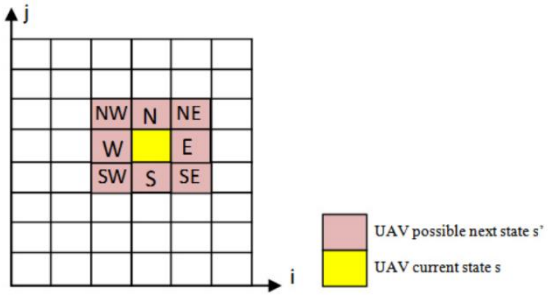


We reduce the problem for this study by addressing a three-dimensional planer problem (movement in three dimensions but not rotation). The three degrees of freedom are represented by the $x - position$, $y - positon$, and heading angle ψ. The height of the UAV $z - position$ will remain constant.
Problem Description: Compute a path for a UAV that uses an unknown, time-varying wind field and minimises energy consumption given two points (Start and Target points). MDP's parameters are as follows:

1) Possible states (S):
   The Cartesian coordinates of the state of the UAV at the centre of a cell will be denoted by $S_{i,j} = x_{i,j},\ y_{i,j},\ \psi_{i,j}$ where $x_{i,j}$, $y_{i,j}$, $\psi_{i,j}$ denote x position, y position and heading angle for the UAV at $cell_{i,j}$ respectively. Velocity of the aircraft is constant and equal to the Minimum Level-Flight Speed (V_{min}).

2) Actions available from each state (A):
   We assume that the UAV can move in eight directions, $A = N, NE, E, SE, S, SW, W, NW$.

3) Transition probabilities (P):
   The transition probabilities P:
   $P_{s,a}(s, s')$ manage the probabilities of what state s' is entered after executing



each action A from state s.
A method based on Gaussian distribution to assign a realistic transition probabilities $P_{s,a}(s,s')$ in a time varying wind field to fit inside the MDP framework was used.
The time-varying wind field is approximated by a Gaussian distribution, at each time step a vector is chosen from the distribution to find the direction and magnitude of the wind field

$$P : P_{s,a}(s,\grave{s}) = \frac{1}{\sigma\sqrt{2\pi}} \int_{\theta_a - \frac{\pi}{8}}^{\theta_a + \frac{\pi}{8}} e^{-\frac{1}{2}(\frac{\upsilon-\omega}{\sigma})^2} d\upsilon.$$

The Standard deviation will be selected by the user and is constant

4) Reward for each transition and each action (R):
The ratio between the wind component facing the target point and the maximum expected wind value will be calculated and multiplying the result by a weight (C) - where (C) is selected by the user.

$$R_a(s_{i,j}) = (\frac{W_{i,j}\cos(\theta_{i,j} + \theta_T)}{W_{max}})C.$$

to this 2C was added if the UAV was in its target state and -5 was added in all other states.

4. Approach Taken

The value function ($V\ (s)$) for a cell will be equal to,

$$V(s_{i,j}) := E[R_a(s_{i,j}) + \gamma \sum (P_{s,a}(s,\grave{s})V(\grave{s})]$$

The optimal value function ($V^*(s)$) for a cell will be given by,

$$V(s_{i,j}) := max_a E[R_a(s_{i,j}) + \gamma \sum (P_{s,a}(s,\grave{s})V(\grave{s})]$$

where s is the initial state, s' the next possible state, $R_a(s_{i,j})$ is the possible reward in state $s_{i,j}$ taken an action a, $P_{s,a}(s,s')$ is the probability of reaching s' while applying action a in state $s_{i,j}$, and $V\ (s')$ is the value function for state s'.

Identifying the optimal values $V^*(s)$ will lead to determining the optimal policy $\pi^*(s)$ using,

$$\pi^*(s) = arg\ max_a(R_a(s_{i,j}) + \gamma \sum_{\grave{s}\in S} (P_{s,a}(s,\grave{s})V^*(\grave{s})$$

The value Iteration algorithm was implemented to generate the optimal policy.

Value Iteration, for estimating $\pi \approx \pi_*$

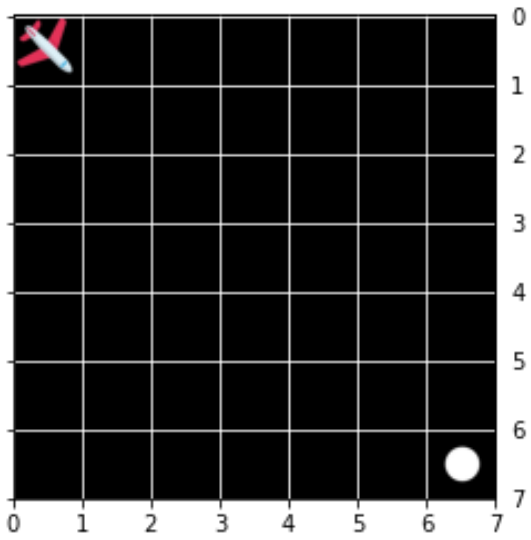
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $v \leftarrow V(s)$
| $V(s) \leftarrow \max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$
| $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
$\pi(s) = \arg\max_a \sum_{s',r} p(s', r \mid s, a)\left[r + \gamma V(s')\right]$

With the help of libraries such as numpy and scipy, all the necessary functions related to the environment were implemented in the Windfield_Env.py file while the value iteration algorithm and the play policy function were implemented in the Agent.py file.

5. Results

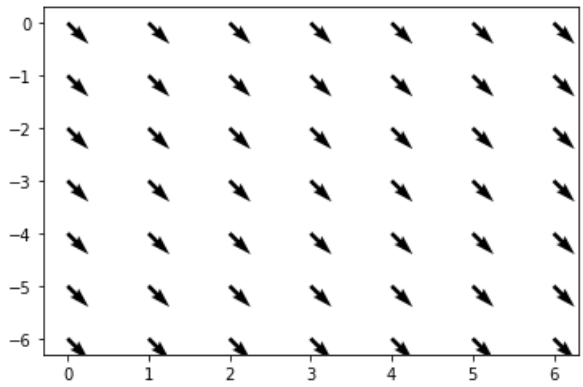


All the examples used a 7x7 grid to represent the wind field.
The starting state is the state (0,0,3) which means its position is (0,0) and heading angle is 3 (corresponds to the SE direction) and the terminal states are the set {(6,6,2),(6,6,3),(6,6,4)} because the position of the terminal state is (6,6) and you can only reach it on taking action 2,3 and 4 at positions (5,6),(5,5) and (6,5) respectively . The Starting state is denoted by a white triangle while the terminal state is denoted by a white circle.

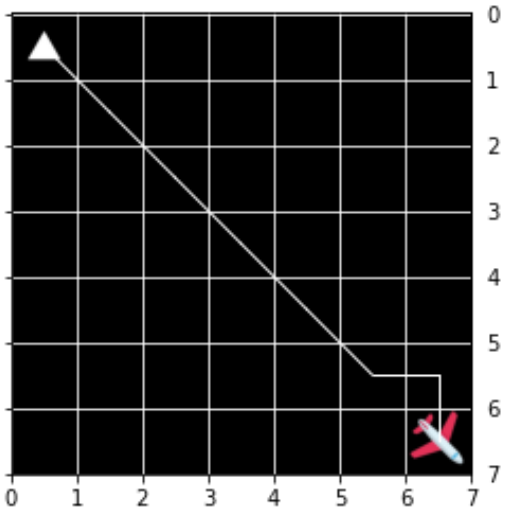
The Algorithm was implemented for 4 different types of wind fields.

1) Wind field Type 1:

The wind field was generated by taking a random vector from a normal distribution of vectors whose means at each state followed the vector field:

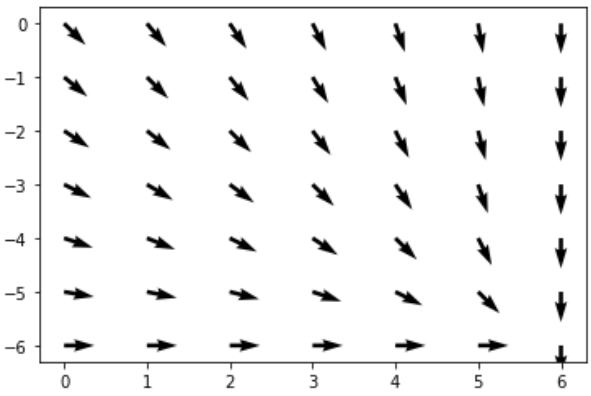


The Algorithm output the following optimal path:

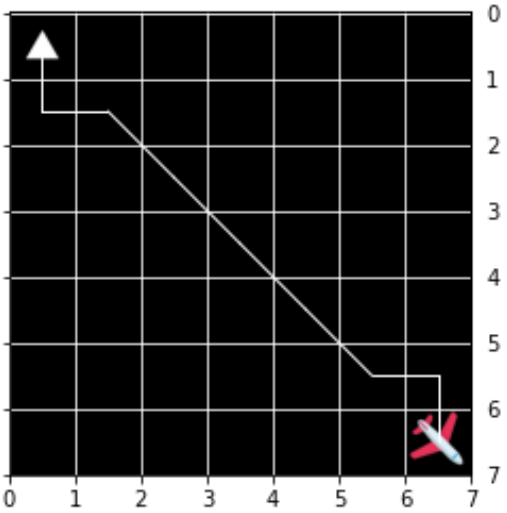


2) Wind field Type 2:

The wind field was generated by taking a random vector from a normal distribution of vectors whose means at each state followed the vector field:

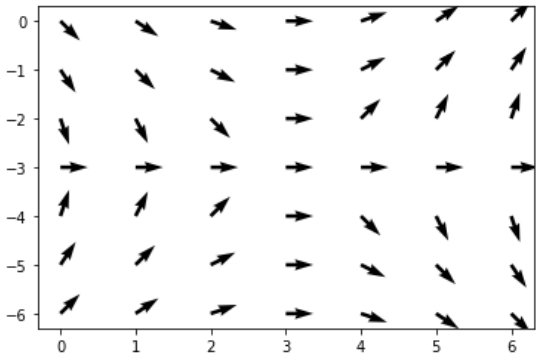


The Algorithm output the following optimal path:

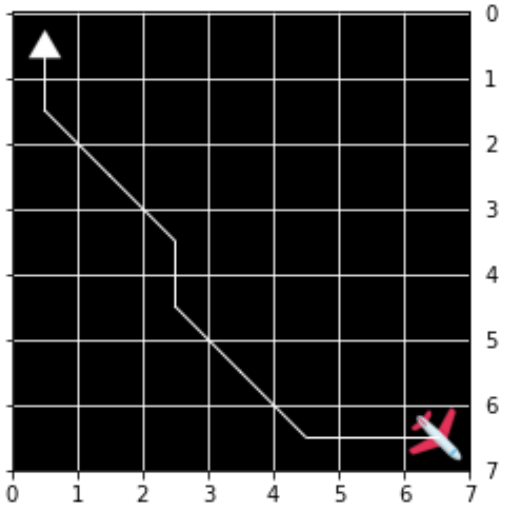


3) Wind field type 3:

The wind field was generated by taking a random vector from a normal distribution of vectors whose means at each state followed the vector field:

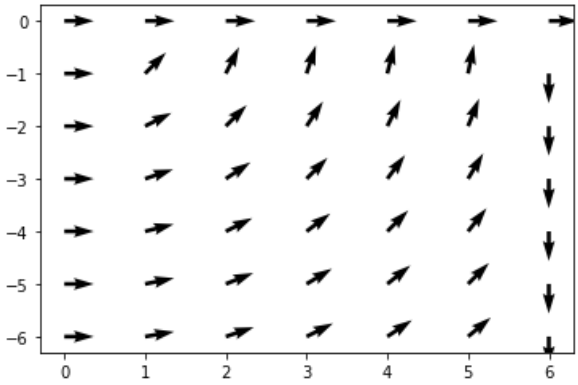


The Algorithm output the following optimal path:

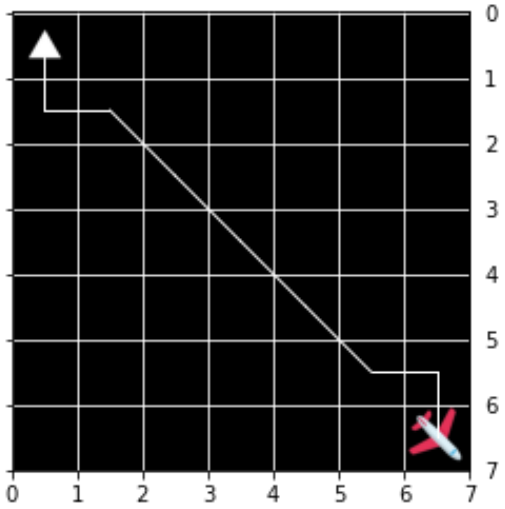


4) Wind field Type 4:

The wind field was generated by taking a random vector from a normal distribution of vectors whose means at each state followed the vector field:



The Algorithm output the following optimal path:



The algorithm seems to have output the optimal path in case of wind field 1, 2 and 3 but the path in case of wind field 3 does not seem to be the most optimal one.

This may be because the -5 reward given for each step may be overpowering the reward due to movement in direction of wind and causing the episode to try to terminate as quickly as possible. This causes the agent to not take longer paths even if it consumes less fuel in doing so.