ENPM667 Project 1

Review of Potential Field Controller on Aerial Robots

Group Members: Piyush Goenka, Yoseph Kebede

UNIVERSITY OF MARYLAND

FEARLESSLY FORWARD

Presentation Outline

1. Introduction
2. Approach
3. Implementation
4. Conclusion

Introduction

Unmanned Autonomous Systems in real-world application

Unmanned Autonomous Systems

Used on ground or on air to assist humans accomplish difficult tasks

- *Wild land fire fighting*
    - Search and rescue operations
    - Military Operations
    - To simplify repetitive work
    - For agricultural work, inspection ...

[5]

[1]

[4]

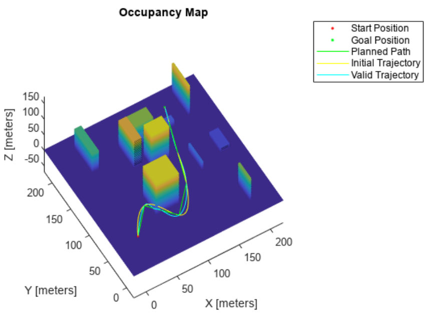
[3]

[2]

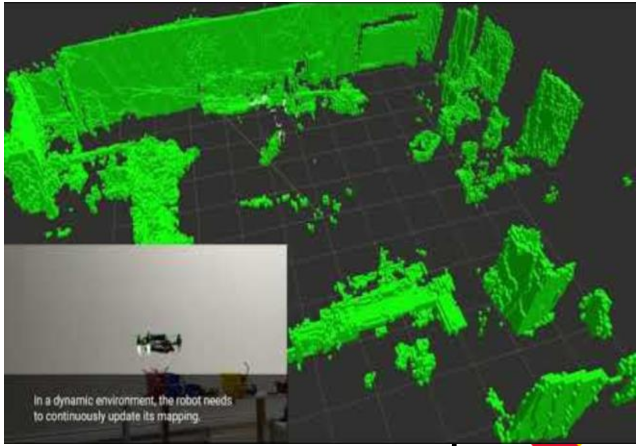
Challenges

- Most Unmanned Autonomous Systems (UAS) are dependent on GPS for real world application.
- Working in GPS-denied regions with UAV has made navigation and localization challenging.
- Attempted Solutions:
  - Using LIDAR to simulate environment with SLAM
  - Using computer vision (like cameras) to visually sense objects in environment to localize
  - Generate smooth trajectories via
    - Minimizing snap via position frame and yaw for smooth path
    - Voronoi diagrams, receding horizons in unrestricted environment, high order parametric curves, 3D-interpolation
- Algorithms require high memory, computation resource than hardware on UAVs
- Aerial robots are more unstable than land

[6] UASs in GPS-Denied Regions



[7] Minimizing Snap Traj Generating



[8] LIDAR and SLAM to navigate and localize

Problem Statement

- Design an efficient, computationally inexpensive architecture
- Solution should be quick to respond to environmental change
- Easy to deploy with proper sensor capabilities

Proposed Solution

- Implement Potential Field Functions to draw vector field map of environment
- Use relative distance and relative velocity as input to define Potential Function and obtain gradients
- Compute velocity functions to track dynamic and stationary obstacles and targets

Approach

Controller Architecture

Assumptions

- UAS is an aerial robot performing in small area, Earth's curvature ignored

- Drone is a quadcopter

- Drone's change in pitch and roll taken too small, ie ignored

- Effect of drag is ignored

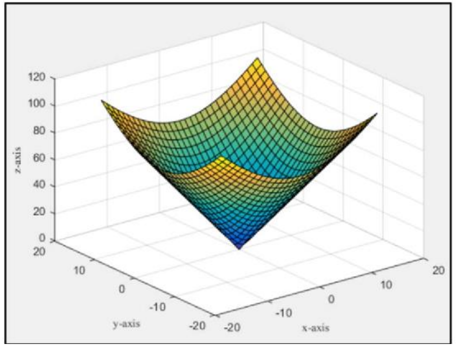
[13] ARDrone 2.0

Traditional Potential Field Controller (PFC)

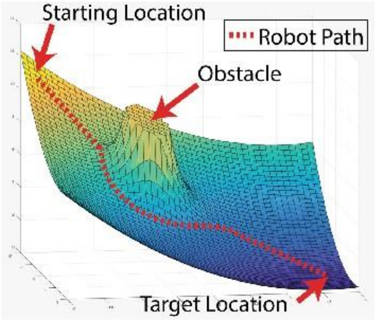
- Define a convex potential function to express the environment, taking relative position between:
- Drone and target (attractive field)
- Drone and obstacle (repulsive field)

$$U_{att^1}(\vec{p_{DT}}) = \frac{1}{2}\lambda_1 ||\vec{p_{DT}}||^2$$

$$U_{rep^1}(\vec{p_{DO}}) = \frac{1}{2}\eta_1 \frac{1}{||\vec{p_{DO}}||^2}$$



[9] Convex potential function



[11] Track target avoid obstacle

- Negative gradient of potential function taken to reach global minimum

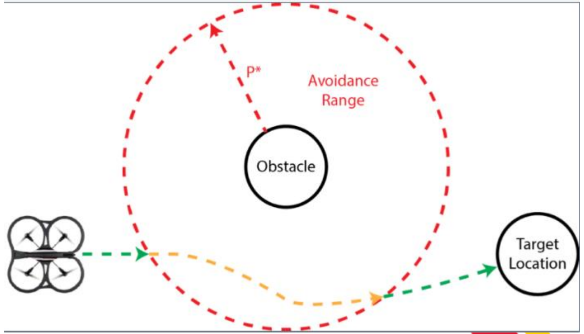
$$\vec{v_{DT}}^{att^1} = -\lambda_1(\vec{p}_{DT})$$

$$\vec{v_{DO}}^{rep^1} = \eta_1 \frac{\vec{p}_{DO}}{||\vec{p}_{DO}||^4}$$

- Define a constant radius for (n) obstacles which drone should avoid

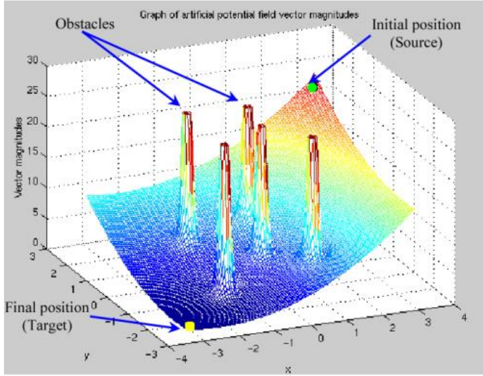
- Drone feels repulsive velocity when in range, and only gets attracted to target when not

$$\boxed{\vec{v}_d^{\mathrm{PFC}}(\vec{p}_d, \vec{p}_t, \vec{p}_o) = \begin{cases} -\lambda_1(\vec{p}_t - \vec{p}_d) + \sum_{i=0}^{n} \eta_1 \frac{\vec{p}_{\mathrm{do}}^i}{\|\vec{p}_{\mathrm{do}}^i\|^4}, & \|\vec{p}_{\mathrm{do}}^i\| \le P^* \\ -\lambda_1(\vec{p}_t - \vec{p}_d), & \|\vec{p}_{\mathrm{do}}^i\| > P^*. \end{cases}}$$



[10] traditional PFC avoid obstacle

Extended Potential Field Controller (ePFC)



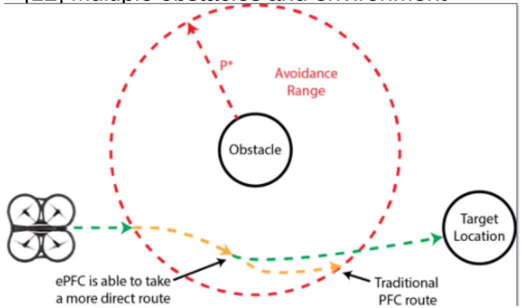
[12] Multiple obstacles and environment

$$U_{att^2}(\vec{v_{DT}}) = \frac{1}{2}\lambda_2 ||\vec{v_{DT}}||^2$$

$$U_{rep^2}(\vec{v}_{DO}) = \frac{1}{2}\eta_2 \frac{1}{||\vec{v}_{DO}||^2}$$

$$\vec{v_{DT}}^{att^2} = -\lambda_2(\vec{v}_{DT})$$

$$\vec{v}_{DO}^{rep^2} = \eta_2 \frac{\vec{v}_{DO}}{||\vec{v}_{DO}||^4}$$



[10] Extended PFC shorter path in avoiding obstacle

$$\vec{v}_{DO}^{rep^2} = \begin{cases} \eta_2 \frac{\vec{v}_{DO}}{||\vec{v}_{DO}||^4}, & ||\vec{v}_{DO}|| \neq 0 \\ 0, & ||\vec{v}_{DO}|| = 0 \end{cases}$$

$$\boxed{\vec{v}_d^{ePFC} = \vec{v}^{PFC} - \lambda_2(\vec{v}_{DT}) + \sum_{i=0}^{n} \eta_2 \frac{\vec{v}_{DO_i}}{||\vec{v}_{DO_i}||^4} - \sum_{i=0}^{n} \eta_3 ||\dot{\vec{p}}_{DO}|| \frac{\dot{\vec{p}}_{DO}}{||\dot{\vec{p}}_{DO}||^4}}$$

$$\boxed{\vec{v}_{d,body}^{ePFC} = \vec{v}_d^{ePFC} * \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \sin(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}}$$

- Starts off by absorbing all principles of Traditional PFC

$$\vec{v}_d^{PFC}(\vec{p}_d, \vec{p}_t, \vec{p}_o) = \begin{cases} -\lambda_1(\vec{p}_t - \vec{p}_d) + \sum_{i=0}^{n} \eta_1 \frac{\vec{p}_{do}^i}{\|\vec{p}_{do}^i\|^4}, & \|\vec{p}_{do}^i\| \leq P^* \\ -\lambda_1(\vec{p}_t - \vec{p}_d), & \|\vec{p}_{do}^i\| > P^*. \end{cases}$$

- Traditional PFC is slow to respond, as it only uses relative distance
- For unstable systems (aerial robots) changing status frequently, fast response is needed
- EPFC adds relative velocity between drone and target , and drone and obstacle for fast response

- The gradient will give the attractive velocity between drone and target, and repulsive velocity with obstacle
- Added repulsive velocity is zero for stationary obstacle
- Path of drone averted when direction of velocity is confirmed to be towards obstacle
- Repulsive velocity halted if drone veers away obstacle, even inside proximity zone
- Final ePFC is sum of traditional PFC and related velocity terms

Stability

- Need to analyze convergence of controller
- Only consider the global function that will achieve goal of guiding rover to target
- EPFC takes relative position and velocity as inputs in its attractive potential function

- A Lyapunov function (L) of the sum of potentials of relative position and velocity is defined (b/c continuous state variables, convex potential functions are decreasing, potential function >= 0)

- System is stable if gradient of L is (-)ve
- Derivaive along p_dt and v_dt is needed, thus the Lie Derivative
- Lie Derivative (L*) is always negative so system (ePFC) is stable

$$L = U_{att} = \frac{1}{2}\lambda_1||\vec{p}_{DT}||^2 + \frac{1}{2}\lambda_2||\vec{v}_{DT}||$$

$$L^{*} = \frac{\partial L}{\partial \vec{p}_{DT}}\frac{\partial}{\partial t}\vec{p}_{DT} + \frac{\partial L}{\partial \vec{p}_{DT}}\frac{\partial}{\partial t}\vec{v}_{DT}$$

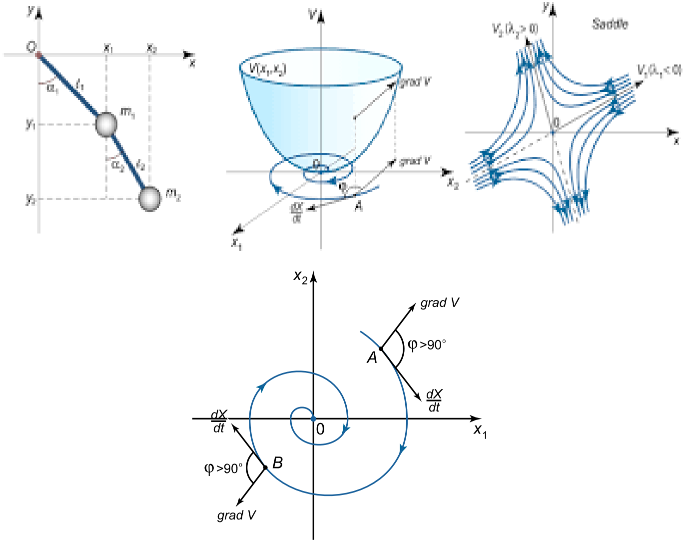
$$L^{*} = \lambda_1||\vec{p}_{DT}||\vec{v}_{DT} + \lambda_2||\vec{v}_{DT}||\vec{a}_{DT}$$

$$\vec{a}_{DT} = \dot{\vec{v}}_{dt}$$

$$\vec{a}_{DT} = -\lambda_2||\tfrac{d}{dt}\vec{v}_{DT}||$$

$$L^{*} = \lambda_1||\vec{p}_{DT}||\vec{v}_{DT} + \lambda_2||\vec{v}_{DT}||\vec{a}_{DT}$$

$$\boxed{L^{*} = -[\lambda_1^2||\vec{p}_{DT}||^2 + \lambda_2^2||\vec{v}_{DT}|| * ||\tfrac{d}{dt}\vec{v}_{DT}||]}$$



[14] Lyapunov Function for two variable system

Implementation

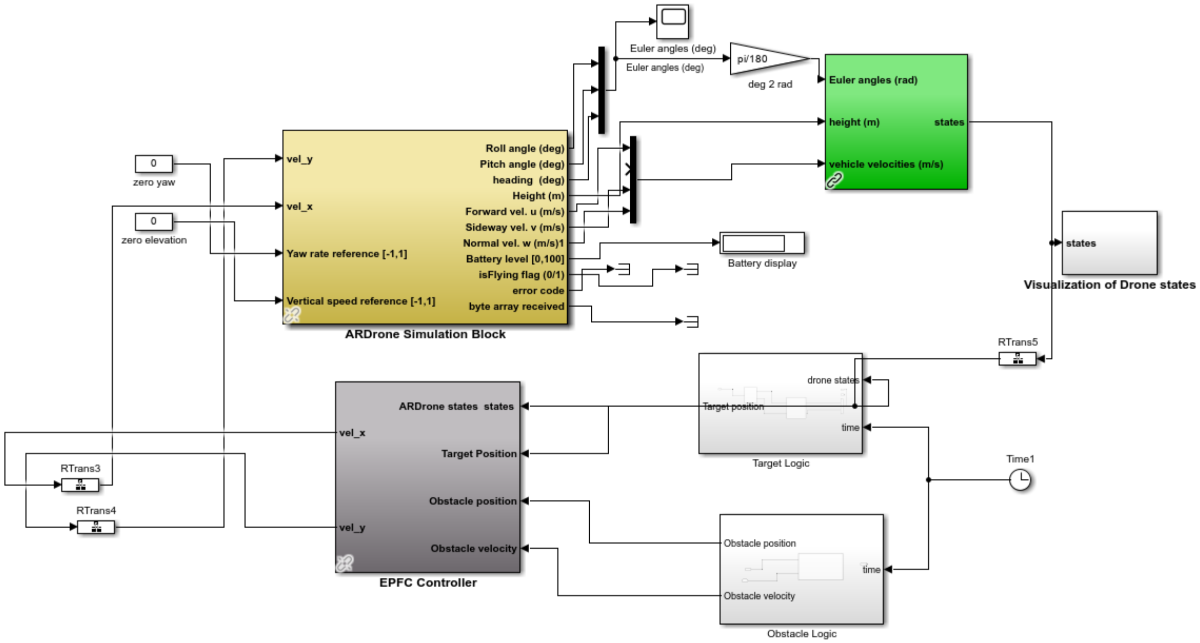
MATLAB and ROS/Gazebo

Simulation Setup

We used a MATLAB Simulink environment to perform the simulation.

The AR Drone Simulink Development Kit v1.1 Add-In for MATLAB was used to obtain the Simulink model of the AR Drone’s state space representation.

We created our custom PFC/EPFC controller model along with our Target and Obstacle logic.



ePFC controller + AR Drone Simulink model

Simulation Results (1/2)

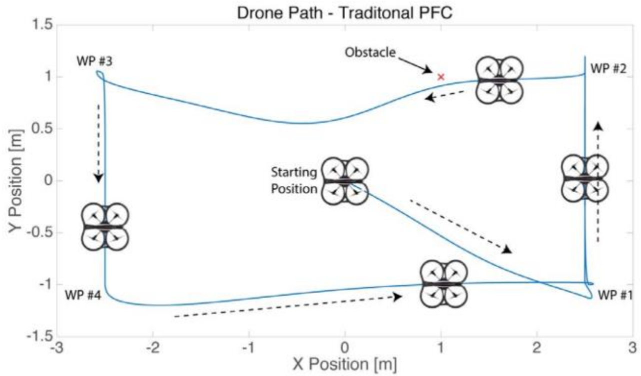
We conducted the simulation for the same waypoint and obstacle configuration as in the original paper.

After numerous testing iterations, λ1 = 0.2 and η1 = 0.03 were used for the PFC controller. While, for the ePFC controller, we used λ2 = 0.05, η2 = 0.01 and η3 = 2.5.

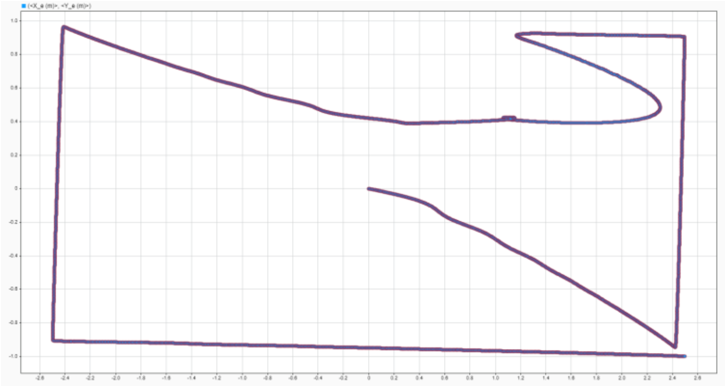
Simulation of PFC Controller:

By comparing the trajectories of the AR Drone using the PFC controller, we can see that both in the original paper and in our simulation, the drone overshoots its path and has a high repulsive reaction to the obstacle.

PFC controller in original paper



PFC controller in our simulation Sim Results



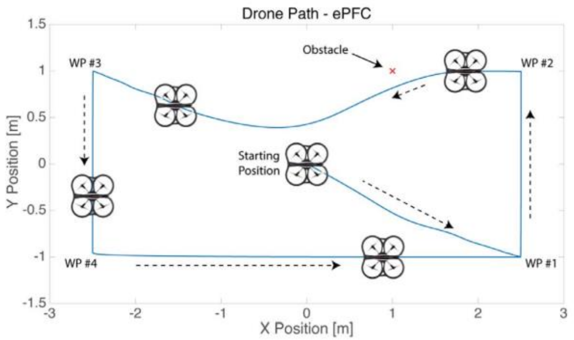
Simulation Results (2/2)

Simulation of PFC Controller:

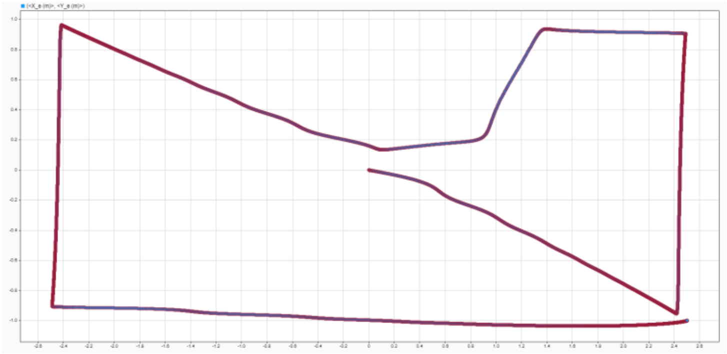
Comparing the trajectories of the AR Drone using the PFC controller, we notice that the trajectory is smooth and the repulsive reaction to the obstacle is handled much smoothly.

Overall, comparing the trajectories of the PFC and the ePFC in our simulation, we can clearly notice that ePFC outperforms PFC with less overshoot and smooth trajectory amid the obstacle's presence.

ePFC controller in original paper



ePFC controller in our simulation Sim Results

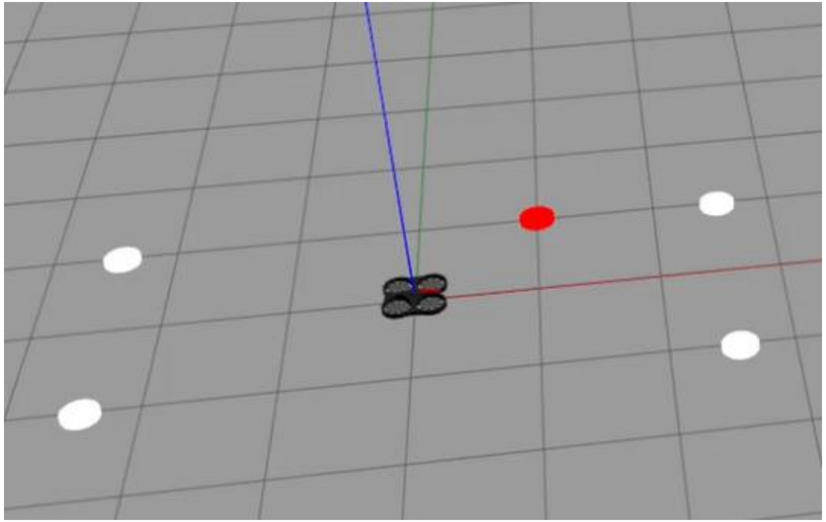


Experimental Setup

Since we did not have the access to the ARDrone hardware, we performed our experiments in the Gazebo simulator in conjunction with the Robot Operating System (ROS) framework. We specifically used ROS Noetic.

We conducted **six** experiments and for each experiment we have a different combination of waypoints and obstacles. Hence, we created six gazebo worlds with white tablets indicating the position of waypoints/targets and red tablets indicating the position of obstacles.

We employed the sjtu-drone ROS package to effectively simulate the AR Drone in the Gazebo worlds that we created.



Gazebo world *(waypoint_1.world)*

GAZEBO

ROS
Robot Operating System

Experimental Results

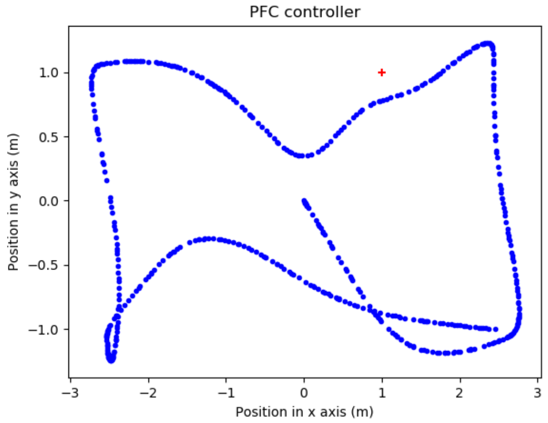
- Experiment 1

The first experiment is setup as identical to the Simulink simulation setup with four targets around the drone and one obstacle in its way.

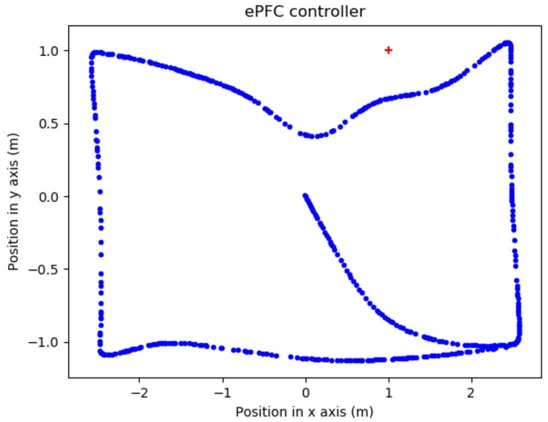
First, we simulate the traditional PFC with $\lambda1 = 1.4$ and $\eta1 = 0.032$. The resulting trajectory can be observed on the right.

The extended PFC is simulated with $\lambda1 = 1.4$, $\eta1 = 0.035$ and $\lambda2 = 0.28$. The resulting trajectory can be observed on the right.

Comparing the trajectories of the drone using both the controllers, we can clearly see that the ePFC performs better than the PFC because it has less overshoot and hence ends up taking a smaller path, which reduces the time of flight while also ensuring that the drone does not go off-course. Also, the ePFC has a higher margin in avoiding obstacles as compared to the traditional PFC.



Sim Results



Sim Results

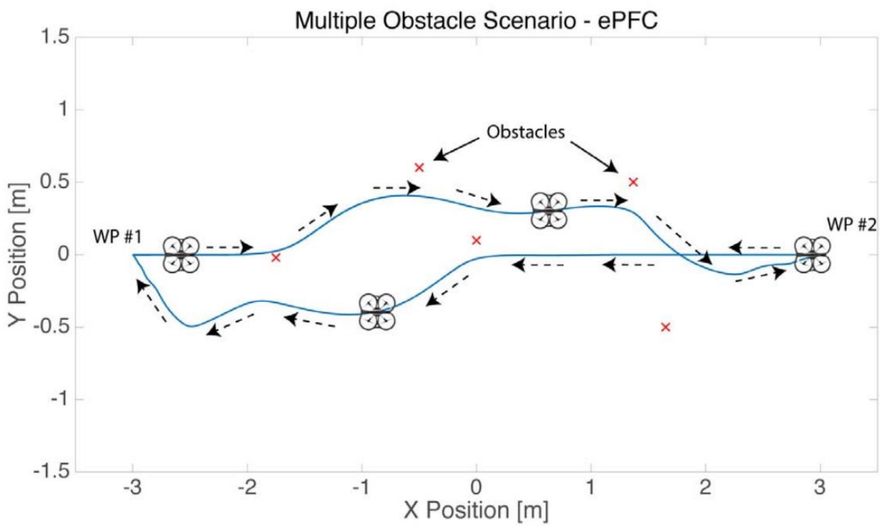
Experiment Results

- Experiment 2 (1/2)

Now, we perform a more complex comparison of both the controllers. In this scenario we put multiple obstacles on the drone's path and take note of how both the controllers perform.

While the original paper conducts the experiment with only the ePFC for a case of multiple obstacles present in the scene, we extend the scenario to compare between the traditional PFC and the ePFC.

The traditional PFC is simulated with $\lambda 1 = 1.0$ and $\eta 1 = 0.029$. The extended PFC is simulated with $\lambda 1 = 1.0$, $\eta 1 = 0.029$ and $\lambda 2 = 0.28$.

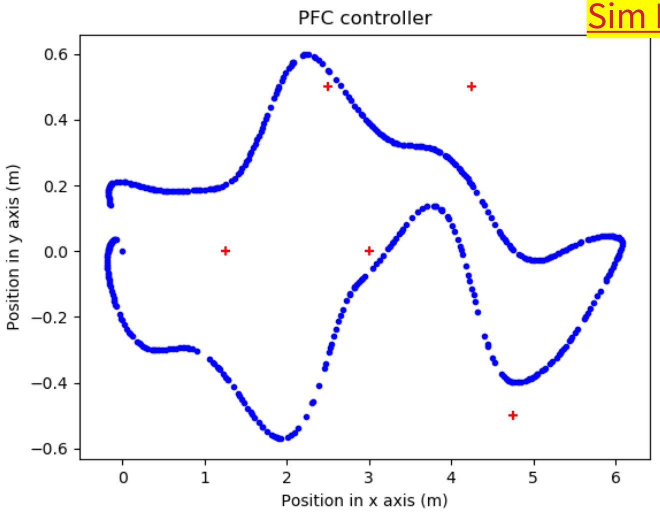


Multiple Obstacle Scenario in original paper

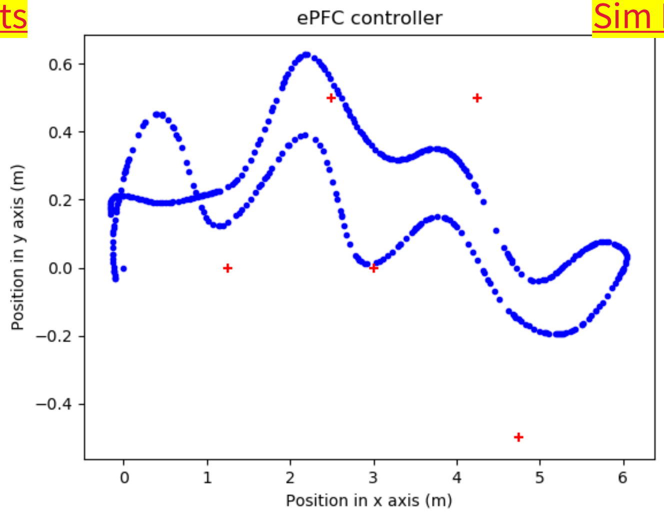
Experiment Results

- Experiment 2 (2/2)

Comparing both the trajectories, we can conclude that the ePFC takes a smaller path and is much more stable and effective in a multi-obstacle scenario.



Sim Results



Sim Results

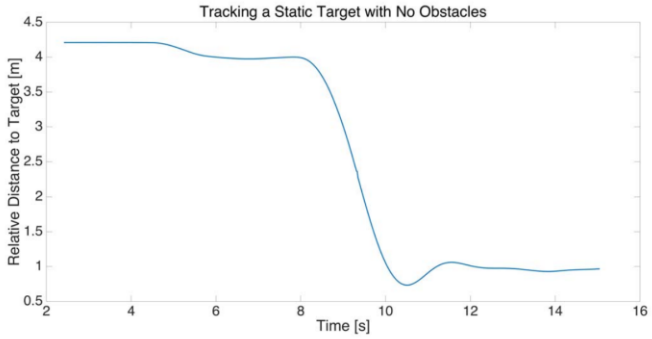
Experiment Results

- Experiment 3

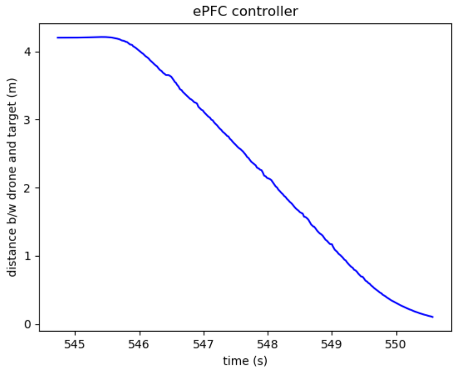
In this experiment, we are interested in observing the nature of change in relative distance between the drone and the target over time. Here, we only have one waypoint and no obstacles.

In our experiment we found that the drone takes 5.5 seconds (t=545 to t=550.5) to reach the target. This value is quite close to the one in the original paper (5 seconds).

Static Target tracking in original paper



Static Target tracking in our experiment



Sim Results

Experiment Results

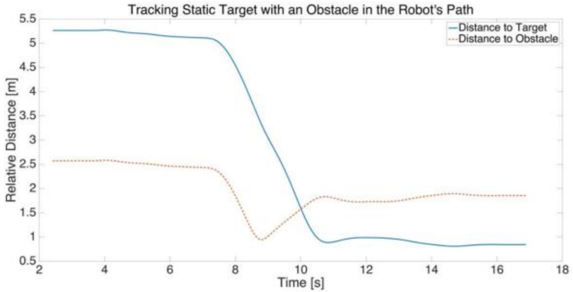
- Experiment 4

In this experiment, we are interested in finding out the change in relative distance between the drone and target as well as between the drone and the obstacle.

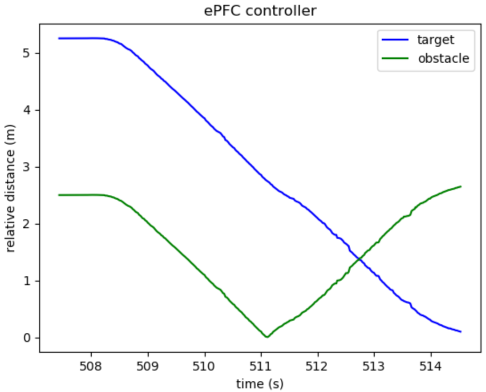
Comparing the results obtained in the original paper with the one we obtained in our experiment we see that the nature of the graph obtained by us is similar to the one in the original paper.

Hence, the controller behaves as expected.

Static Target tracking with obstacle in original paper



Static Target tracking with obstacle in our experiment



Sim Results

Experiment Results

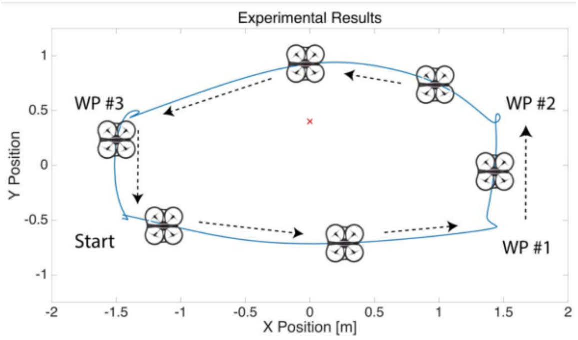
- Experiment 5

In this experiment we test the controller on a setup similar to that of the simulation but with closer waypoints and an obstacle.

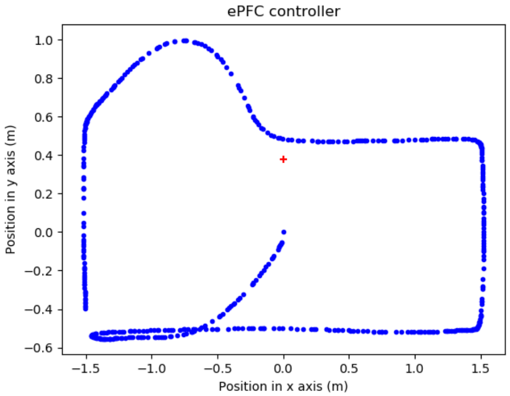
We notice the drone's trajectory in the original paper and observe that the trajectory obtained from our experiment is not similar.

However, we also observe that the controller is effective in avoiding the obstacle in its path and is able to traverse between waypoints without overshoot.

AR Drone trajectory in original paper



AR Drone trajectory in our experiment



Sim Results

Experiment Results

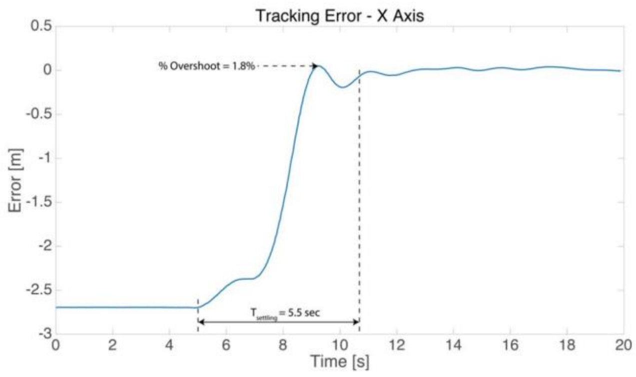
- Experiment 6

In this experiment, we are interested in finding out the error in the trajectory. Here we have a waypoint which is at a considerable distance from the drone in the x-direction (2.7m) and also is offset the y-direction (1m). We are interested in observing the tracking error in the x-direction.

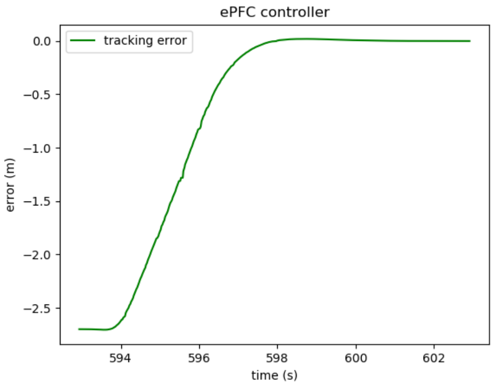
We observe that the settling time for the drone in our case is 6s (t=594 to t=600) which is close to the one in the original paper of 5.5s.

Here we also observe that there isn't much of an overshoot as observed in the original paper. This particular observation highlights the existing sim2real gap.

Tracking error in original paper



Tracking error in our experiment



Sim Results

Conclusion

Challenges, Reflections and Acknowledgements

Conclusion

The paper under review introduces an extended Potential Field Controller (ePFC), which enriches the traditional PFC by incorporating feedback from relative velocities between a drone and its target or obstacles.

The stability of this controller is rigorously established using Lyapunov methods. Through MATLAB simulation, the paper evaluates the ePFC's performance in comparison to a standard PFC, demonstrating its superiority in reducing overshoot and settling time during waypoint navigation.

The inclusion of experimental findings provides tangible evidence of the controller's effectiveness in real-world scenarios. The repeatibility of the simulation and experiments helped validate the controller's effectiveness. From our experiments we observed that ePFC is an ideal candidate for aerial robots.

Since we were not able to perform our experiments on the real drone, our future work may include testing the controller on the real AR Drone and possibly extending it to other drones.

Team Contribution

1. Piyush Goenka
   - Conducted MATLAB/Simulink simulation
   - Conducted experiments in ROS/Gazebo

2. Yoseph Kebede
   - Conduct Literature Review
   - Perform derivation of modeling drone, controller, and stability

Thank you to Dr. Malik and TAs

In slide References

1. https://interdrone.com/public-safety/oas-requests-information-for-heavy-lift-uas-during-wildfires/
2. https://www.militaryaerospace.com/uncrewed/article/14291184/unmanned-bomb-disposal-ground-vehicles
3. https://hdiac.org/articles/communicating-in-remote-areas-or-disaster-situations-using-unmanned-aerial-vehicles/
4. https://robotnik.eu/how-does-mobile-robot-technology-increase-warehouse-productivity/
5. https://www.farmersweekly.co.za/agri-news/south-africa/agri-robotics-set-to-take-off-in-south-africa/
6. https://uavcoach.com/gps-denied-drones/
7. https://www.mathworks.com/help/uav/ug/plan-minimum-snap-trajectory-for-quadrotor.html
8. https://www.youtube.com/watch?app=desktop&v=EbzfjNpLzvg
9. Galvez, Reagan L. et al. "Obstacle avoidance algorithm for swarm of quadrotor unmanned aerial vehicle using artificial potential fields." *TENCON 2017 - 2017 IEEE Region 10 Conference* (2017): 2307-2312.

References (cont...)

10. A. C. Woods and H. M. La, "A Novel Potential Field Controller for Use on Aerial Robots," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 49, no. 4, pp. 665-676, April 2019, doi: 10.1109/TSMC.2017.2702701.
11. Woods, Alexander & La, Hung. (2015). Dynamic Target Tracking and Obstacle Avoidance using a Drone. 10.1007/978-3-319-27857-5_76.
12. Hargas, Y., Mokrane, A., Hentout, A., Hachour, O., & Bouzouia, B. (2015). Mobile manipulator path planning based on artificial potential field: Application on RobuTER/ULM. 2015 4th International Conference on Electrical Engineering (ICEE), 1-6.
13. https://www.amazon.com/Parrot-AR-Drone-2-0-Elite-Quadcopter/dp/B00FS7SU7K
14. https://math24.net/method-lyapunov-functions.html