Fall 2023

ENPM667 Project 1

Review of Potential Field Controller on Aerial Robots

November 14, 2023

Students:
Piyush Goenka, 120189500

Yoseph Ayele Kebede, 114196729

Instructors:
Dr. Waseem Ansar Malik

Course code:
ENPM667

Review of Potential Field Controller on Aerial Robots

Piyush Goenka
A. James Clark School of Engineering
University of Maryland College Park
College Park, Maryland
Email: pgoenka@umd.edu

Yoseph Ayele Kebede
A. James Clark School of Engineering
University of Maryland College Park
College Park, Maryland
Email: ykebede2@umd.edu

***Abstract*—In recent years, the use of of autonomous robots, both ground and aerial has been indispensable means of support for people working in various sects of life for both mundane tasks and emergencies. One common task of using these robots, or unmanned autonomous vehicles, has been in the task of performing tracking objects which require accurate algorithms to navigate their surrounding and plan their paths efficiently. Of the two, aerial robots pose higher challenge as they are more unstable compared to their ground counterparts. Thus, out of the various approaches deployed by researchers, this paper reviews implementation of a simpler and lighter technique (both in steps and hardware demand) called extended Potential Field Controller that shows promising results for designing a smart tracking drone that avoids obstacles. Furthermore, the derivations and experimental verification are demonstrated fully to assess the authors' assertions**

***Index Terms*—Aerial robots, drones, unmanned autonomous systems (UAS), unmanned autonomous vehicles (UAV), rover**

I. Introduction

The advent of robots and robotic systems has brought multitudes of benefits to various industries from emergency services such as firefighting as shown in Fig. 1 and human extraction from underneath rubble to military operations and security all the way to day to day tasks such as warehouse operations [1]. Therefore, as these robots keep demonstrating heightened performance in the coming years, their dependability is increasing as well which expose these robots to ever more complex settings. Some of these tasks include search and rescue [2] where robots engage in environmental sensing as well as transporting essential packages to the active site; goods and merchandise delivery [3] where robots on ground or on air need to navigate around obstacles and deploy products at a destination; and robotic swarm deployment during military operations [4] that require robots to not just know their locations and that of their goal, but the other robots in their 'team' plus various environmental conditions. In all of these, and several other tasks requiring robots covering vast space in a working control volume, drones are becoming desirable [5] due to their simple maneuverability and ability to move in confined spaces.

Despite their popularity in different sectors, drones (similar to ground robots) have encountered limitations in navigation and localization in regions that could not have access to or require more resolution than GPS [7]. As a result, several methods have been proposed to tackle this challenge. The prominent solution thus far is sensing the environment the robot is operating on devices generating light detection and ranging (LIDAR) data where teams [8] have developed algorithms responding to this data in uncertain environments. Similarly, others [9] utilize multilevel simultaneous localization and mapping (SLAM), leveraging LIDAR data to create a second sense of 'vision' and plan subsequent paths of robot, while autonomous navigation has also been testing using these methods for multi-floor operations [10], [11]

Fig. 1. FireFighter using drone for training. Application of quadcopter drone in real life scenarios.
[6]

Other approaches to get around lack of GPS has been via use of techniques in computer vision where images of the surrounding are manipulated to provide important information of the environment to robot. Some implementations have been using object sizes to determine relative distance of rovers [12], adding smart devices in the loop such as Microsoft Kinect to visualize their environment [13], and others have investigated in using localization principles along with image processing techniques [14], [15] to define the surrounding these robots are placed in.

Once sensing and localization is completed, these autonomous robots are then expected to traverse in these secluded regions to reach the desired destination safely in the most optimal form possible. This means that these machines need to generate optimal trajectories - which account to changes in the environment in real time - that will enable them to avoid barriers as well as obstacles (static and dynamic) to accomplish their mission. In order to solve these challenges, various implementations that use time parameterized polynomial trajectories leading UAVs to attain pre-set waypoints in short time spans [16], minimizing second derivative of acceleration (snap) to use visual-inertial odometry data in generating feasible trajectories [17], [18], utilizing Voronoi diagrams to generate smooth trajectories [12], [19], receding horizons [20] and high-order parametric curves [21] are other promising attempts used to name a few.

Although the results of these implementations is promising, there is one key factor that determines whether they will be used or not on actual robots, for the purpose of this review we will narrow down the scope to drones as was tested on the paper being discussed; that is processing power of hardware on the drones. Thus, most of the techniques mentioned above do not perform their full strength on the battery-time and processing compute limited drones, unless the drone receives off line help from servers or wifi access points [22] which may not be practical for the vast majority of the drone operators. As a result, the authors of the discussed extended Potential Field Controller (ePFC) in this paper, argue that their method is computationally inexpensive and easy to assemble while maintaining high class performance when the drone interacts with the environment in accomplishing its assigned task. The ePFC discussed in detail below, further expands the traditional potential field controller (PFC) - which uses relative distances for obstacle avoidance and target pursuit - by adding relative velocities to achieve similar goals faster.

In this paper, the report on the extended Potential Field Controller is replicated and results thoroughly tested as in the original publication. The subsequent sections are structured similarly where general modeling of the quadcopter (drone) system dynamics is discussed in Section II. In Section III, the design of the potential field methods, the controller, and stability check of Lyapunov approach are demonstrated with their full derivation. Next, controller simulation with design test points are demonstrated using MATLAB in Section IV, with life-like results test environment built using ROS and simulated in Gazebo. Section V will thoroughly discuss results from MATLAB from the various test scenarios, as well as assessment on the controller performance. At the end finishing remarks on the reviewed paper, implementation of ePFC as a whole, and lessons learnt will be presented.

II. Modeling-System

The quadcopter is a system changing in time and spatial units within the control volume it is operating in. This, as a results, leads to the development of series of differential equations to further understand changes in the system, and begin in developing the design of appropriate controllers to model the rover. As a result, initial assumptions need to be made using Newtonian reference frames - to describe the drones reference frame with respect to the world - and define it as a regid body.

A. Reference-Frames

To begin deriving set of mathematical equations describing motion of drone, it is important to set perspectives in defining the location and orientation of the robot with respect to a common reference frame. Hence, starting with the center of mass assumption for a particle, a set of linearly independent vectors representing the 3D world via Cartesian reference frames is fixed on earth, which is considered to be inertial and fixed for the current drone example. Furthermore, since the scope of this research is for indoors GPS-denied regions, the control volume is small enough that curvature of the Earth will not come into play. Therefore, orthogonal Real 3D vector elements are used to represent positive directions of the fixed global frame.

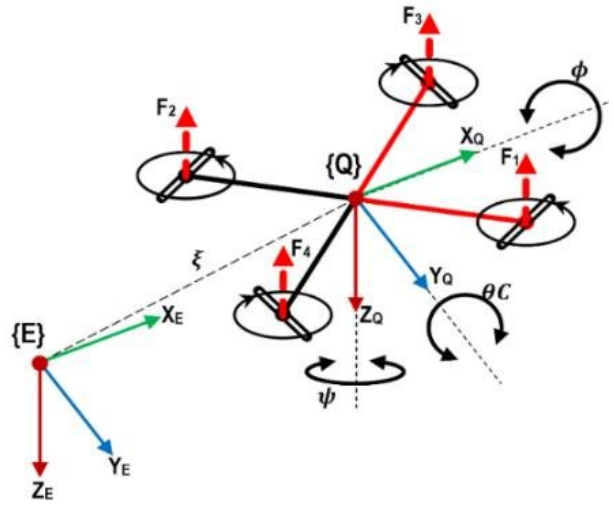

Fig. 2. Inertial reference (world) frame fixed on earth and body frame fixed in a rigin body's (plane) center of mass. Object's position and velocity can be described with respect to fixed frame, while euler angles used to describe object's movement with respect to its own frame.
[23]

As shown in Fig. 2 above, the world reference frame is defined using the general principle of North (X_E), East (Y_E), and Down (Z_E) which points toward the center of the earth. On the other hand the body reference frame - which is noninertial and experiences acceleration - is taken to have the drone's forward direction as X_Q, Y_Q pointing to the right from this attitude, and Z_Q pointing orthogonally in the downward direction. Finally, the angles pertaining to rotations with respect to the drone's frame of reference can be assumed to be ϕ, θ, and ψ which represent roll, pitch and yaw respectively.

B. Euler Transformations

Since the paper in this review aims to prove the efficiency of Extended Potential Field Control which takes relative

position and velocity as an input, it's important to find the transformation between the fixed and body reference frames. The three popular methods discussed in the paper are Euler rotation matrices, quaternion transformations and angle-axis representation. Since Euler transformation is the method eventually pursued, though it has limitations in general overcome by the other methods, it is method discussed below fully along with why limitations do not matter in the analysis presented in this paper.

To begin with, the drone would be away from the origin of the fixed world frame with a certain translation (euclidean distance which can be represented in the NED directions described in the previous section) and rotation (described with roll, pitch and yaw angles). Since obtaining the translation is relatively straightforward, which can easily be placed in an X_E, Y_E, Z_E vector or in a unit vector with computed euclidean distance form, as shown in Fig. 3.

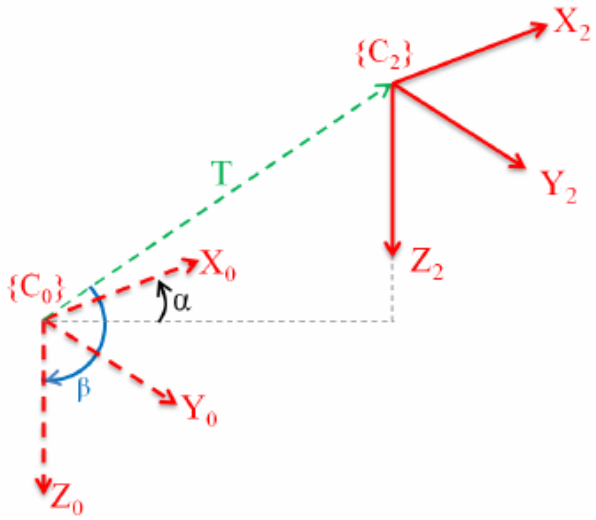

Fig. 3. Determining the translation of a noninertial frame using linearly independent position vectors

[24]

Moving forward, the focus when studying reference on transformations is on rotational motion with the drone rotating about the three major axes in its own frame. First, let's start by rotating the body frame by ψ angles with respect to the Z_Q frame, as shown in Fig. 4.

Then, when looking at the x and y plane alone (as z axis stays the same shown in Fig. 5), we can better define the new coordinate frames x' and y' in terms of ψ as follows.

$$\begin{aligned} e'_x &= \cos(\psi)e_x + \sin(\psi)e_y \\ e'_y &= -\sin(\psi)e_x + \cos(\psi)e_y \\ e'_z &= e_z \end{aligned} \tag{1}$$

Thus, writing these systems of equations describing rotation in matrix form will be equal to:

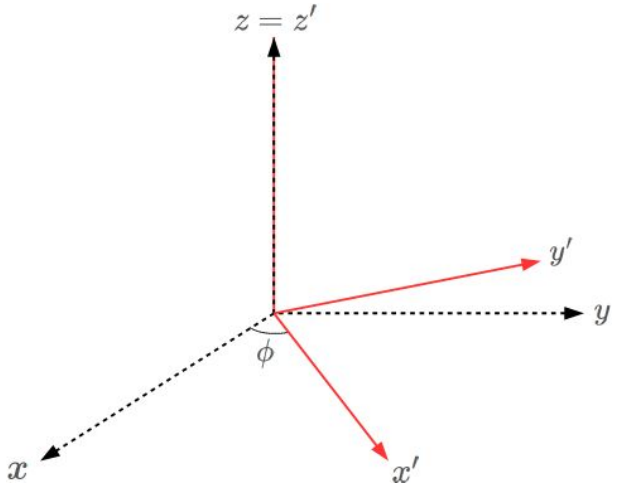

Fig. 4. Determining the rotation of the noninertial frame about the Z_Q axis

[25]

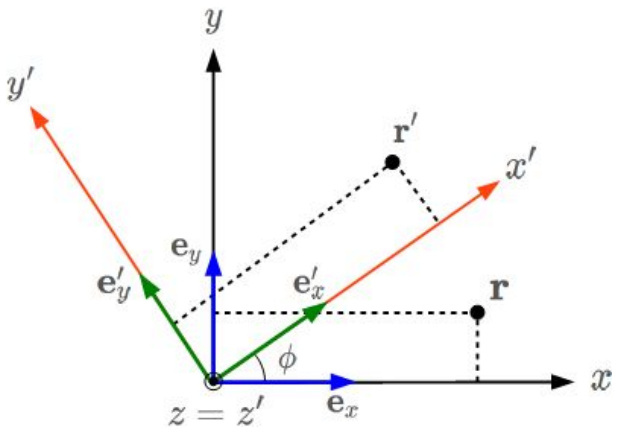

Fig. 5. Determining the rotation of the noninertial frame about the Z_Q axis, top view

[26]

$$R_z = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Similarly, the rotations about the x and y axes will respectively be

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \tag{3}$$

$$R_y = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{4}$$

Now that the three rotation matrices for each of the axes have been define, for ease of use they are combined into general rotation R that can be used to convert from one frame to the other

$$R = R_x * R_y * R_z$$

$$R_y * R_z = \begin{bmatrix} cos(\theta)cos(\psi) & cos(\theta)sin(\psi) & -\sin(theta) \\ -\sin(\psi) & \cos(\psi) & 0 \\ cos(\theta)sin(\psi) & \sin(\theta)\sin(\psi) & \cos(\theta) \end{bmatrix} \tag{5}$$
$$C = \cos(), S = \sin()$$

$$\begin{aligned} R &= R_x * R_y * R_z \\ &= \begin{bmatrix} \mathrm{C}_\theta \mathrm{C}_\psi & \mathrm{S}_\phi \mathrm{C}_\psi \mathrm{S}_\theta \text{-} \mathrm{C}_\phi \mathrm{S}_\psi & \mathrm{C}_\phi \mathrm{C}_\psi \mathrm{S}_\theta \text{+} \mathrm{S}_\phi \mathrm{S}_\psi \\ \mathrm{S}_\phi \mathrm{C}_\theta & \mathrm{S}_\phi \mathrm{S}_\psi \mathrm{S}_\theta \text{+} \mathrm{C}_\phi \mathrm{C}_\psi & \mathrm{S}_\psi \mathrm{C}_\phi \mathrm{S}_\theta \text{-} \mathrm{C}_\psi \mathrm{S}_\phi \\ -\mathrm{S}_\theta & \mathrm{S}_\phi \mathrm{C}_\theta & \mathrm{C}_\phi \mathrm{C}_\theta \end{bmatrix} \end{aligned} \tag{6}$$

Hence, by using the full rotational matrices and additional properties, units in the body frame can be transformed to the world frame, by multiplying the position [x, y, z] vector with these matrices. A downside to this method, is when that cos() term becomes zero which leads to a loss in one dimension, also called gimbal lock. Thus, in our definition here, that translates to the drone making 90°rotations in any of the three axes. Although this is one hindrance of this method which let people use other techniques like quaternions, the authors of the original paper assumed no such large angles and pursued with this approach which is what this review will follow as well. With the frame transformations now defined, studying the net force and torque on the drone next will better help understand the system

C. Defining System with Newton-Euler Equations

Following on the definition of rigid body and their locations with respect to a reference frame, we would use Newton-Euler equations to fully define the linear and transnational motions of the rigid body.

Using the fixed reference frame, the time rate of position $\vec{p}$ of the rigid body is given by

$$\vec{v} = \frac{d\vec{p}}{dt} \tag{7}$$

Then, using Newton's second law where net Force equals derivative of translational momentum $\vec{L}$,

$$\vec{L} = m\vec{v} \tag{8}$$

which is given by

$$\begin{aligned} \sum \vec{F} &= \tfrac{d}{dt}(\vec{L}) \\ &= \tfrac{d}{dt}(m\vec{v}) \end{aligned} \tag{9}$$

Furthermore, since the assumption here is that mass of the drone stays constant, taking out the constant mass variable and deriving the velocity with respect to time we get

$$\begin{aligned} \sum \vec{F} &= m\tfrac{d\vec{v}}{dt} \\ &= m\vec{a} \end{aligned} \tag{10}$$

where a is the linear acceleration of the drone in the fixed world's coordinate frame. Similarly, the drone's movement can be describe from its own frame of reference. Thus, since the drone has a noninertial frame of reference, it's center of mass will have some velocity, in addition to any rotation of the points off from it's center (denoted by r_c) which have a rotational velocity with respect to the center that can be shown as:

$$\begin{aligned} \sum \vec{F}_d &= m\tfrac{d}{dt}(v_t + v_r) \\ &= m(\tfrac{d\vec{v}_{trans}}{dt} + \tfrac{d\vec{v}_{rot}}{dt}) \\ &= m(\vec{a}_d + \tfrac{d(\vec{\omega} * r_c)}{dt}) \\ &= m(\vec{a}_d + \vec{\omega} * \vec{v}_d) \end{aligned} \tag{11}$$

Nonetheless, since the derivation is from the center of mass of the drone where it's fixed frame is centered, the angular velocity ($\vec{\omega}$) is zero, further reducing the sum of of forces equation to

$$\sum \vec{F}_d = m\vec{a}_d \tag{12}$$

In addition to linear motion dynamics, the drone will have angular movements that need to be accounted for which is where the Euler part comes in whose second law equates net Torque to derivative of angular momentum. The angular momentum is defined as

$$\sum \tau_d = \vec{H} = I_{cm} * \vec{\omega} \tag{13}$$

where $\vec{\omega}$ is the angular velocity particle on the drone r distance away from the center of mass (body reference origin) and the moment of Inertia I_{cm} with respect to center of mass for our point mass (r distance away) assumption is

$$I_{cm} = m * r^2 \tag{14}$$

with I_{cm} to taken to be time invariant for current study.

It is important to know that however the drone is a non inertial frame and thus it will be rotating and traversing. Hence, to define the rotation of the points within its body followed by their derivatives, they would need to be expressed in two terms, one from explicit time dependence due to motion of points within the rotating reference frame and the other from frame's own rotation.

$$\begin{aligned} \tfrac{d\vec{I_{cm} * \vec{\omega}}}{dt} &= [(\tfrac{d}{dt} + \omega x](I_{cm} * \vec{\omega}) \\ &= \tfrac{d}{dt}(I_{cm} * \vec{\omega}) + \omega x(I_{cm} * \vec{\omega}) \end{aligned} \tag{15}$$

Therefore, solving Euler's second law would then give

$$\begin{aligned} \sum \vec{\tau}_d &= \tfrac{d}{dt}(I_{cm} * \vec{\omega}) \\ &= \tfrac{d\vec{I_{cm}}}{dt}\omega + I_{cm}\tfrac{d\vec{\omega}}{dt} \\ &= 0 + \tfrac{dI_{cm} * \vec{\omega}}{dt} + \omega x(I_{cm} X \vec{\omega}) \\ &= I_{cm}\alpha + \omega x(I_{cm} X \vec{\omega}) \end{aligned} \tag{16}$$

α coming out to be the angular acceleration of the drone in the rotating frame

The linear and angular acceleration of the drone can be compactly shown in a matrix as

$$\begin{bmatrix} \sum \vec{F}_d \\ \sum \vec{\tau}_d \end{bmatrix} = \begin{bmatrix} mI_3 & 0 \\ 0 & I_{cm} \end{bmatrix} \begin{bmatrix} \vec{a}_d \\ \vec{\alpha}_d \end{bmatrix} + \begin{bmatrix} 0 \\ \vec{\omega} X I_{cm} \omega \end{bmatrix} \tag{17}$$

Note that, because Force has 3 components in the general x,y,z dimension, the Identity matrix I_3 has been added in the matrix multiplying the constant m.

With the system of equations now defined via the drone's center of mass, next we resume towards setting up its the equations of motion.

D. Quadcopter Drone Dynamics Review

The drone configuration in studied in the ePFC paper was assumed to be a quadcopter with four thrusters placed symmetrically with respect to the drone's x and y axes. This, as a result, simplifies the derivation by setting parameters as symmetric, such as the moment of inertia in the inertia matrix given as,

$$I_{cm} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{18}$$

with I_{xx}, I_{yy}, and I_{zz} denoting the moment of inertia in the principal axes of the drone. First, we would like to compute the sum of the forces on the drone that is hovering on air as shown in Fig. 6, i.e. in motion.

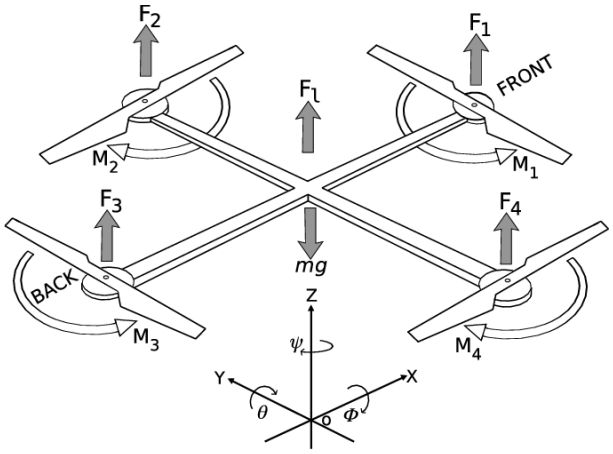

Fig. 6. Quadcopter has symmetric thrusters producing force and moment due to each actuator's angular velocity [27]

The force (F) produced by the propellers generating upward thrust is given by $F = K_f * \omega^2$, where ω is propeller angular speed and K_f is a constant factor depending on back EMF, air density, propeller area, and the like. Since the paper in review assumes constant air density and relatively small (neglected) pitch and yaw motions, the only force term for the analysis of the drove in question would be the sum of all the four identical propellers thrust component, ie

$$\begin{aligned} F_x &= 0 \\ F_y &= 0 \\ F_{z_i} &= K_f * \omega_i^2 \\ \sum F &= mg - 4 * K_f * \sum \omega_i^2 \end{aligned} \tag{19}$$

Note, with down being the body frame positive convention, results in weight (mg) taken as positive. Additionally, each of the propellers generate moments, given by $M = K_m * \omega^2$, that cause the drone to rotate in an opposite direction. Furthermore, since the propellers are aligned on either x or y axes, the two in x would produce equal but opposite moments to the other two propellers on the y. Moreover, since the propellers will not contribute to moments on the axis that they are attached to, the opposite sets are utilized. In sum, moments come to be

$$\begin{aligned} M_x &= (F_2 - F_4) * L \\ M_y &= (F_3 - F_1) * L \\ M_z &= -(M1 - M2 + M3 - M4) \end{aligned} \tag{20}$$

with M1 and M3 producing CCW moment, M2 and M4 CW, and sign for M_z flipped to accomadate for body frame z axis definition. Then, putting Newton and Euler end results concisely, we get

$$\begin{aligned} \sum \vec{F} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & mg - 4 * K_f * \sum \omega_i^2 \end{bmatrix} \begin{bmatrix} x_d \\ y_d \\ z_d \end{bmatrix} \\ \sum \vec{\tau} &= \begin{bmatrix} k_f(\omega_2^2 - \omega_4^2) & 0 & 0 \\ 0 & k_f(\omega_3^2 - \omega_1^2) & 0 \\ 0 & 0 & -k_m(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} \\ &* \begin{bmatrix} x_d \\ y_d \\ z_d \end{bmatrix} \end{aligned} \tag{21}$$

Finally, we can now rewrite the linear and angular accelerations based off of 18 and 21. Since roll and pitch are assumed zero, linear acceleration would be in the z axis with net Force divided by mass of the drone. On the other hand, the angular acceleration is computed rearranging Euler's second law for rigid body dynamics depicted on 18 which comes out to be $\tau = I\alpha$ and $\tau = Moment - \omega x(I\omega)$, which combined give $\alpha = \frac{1}{I}(M - \omega x(I_{cm}\omega)$ where α, ω, I_{cm} and M are in the three body axes of the drone. Expanding the matrix multiplications thoroughly yields,

$$\omega = \begin{bmatrix} \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T$$

$$I_{cm} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

$$\begin{aligned} \alpha &= \tfrac{1}{I}(M - \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} X(\begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}) \\ &= \tfrac{1}{I}(M - \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} x(\begin{bmatrix} I_{XX}\dot{\phi} \\ I_{yy}\dot{\theta} \\ I_{zz}\dot{\psi} \end{bmatrix}) \\ &= \tfrac{1}{I}(M - \begin{bmatrix} \dot{\psi}\dot{\psi}(I_{zz} - I_{yy}) \\ \dot{\psi}\dot{\phi}(I_{xx} - I_{yy}) \\ \dot{\phi}\dot{\theta}(I_{yy} - I_{xx}) \end{bmatrix} \end{aligned} \tag{22}$$

Hence, we can now put the full six degree of freedom motion of the drone as,

$$\begin{bmatrix} \vec{a}_x \\ \vec{a}_y \\ \vec{a}_z \\ \vec{\alpha}_x \\ \vec{\alpha}_y \\ \vec{\alpha}_z \end{bmatrix} = \begin{bmatrix} \frac{1}{m} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{m} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix}$$
$$X(\begin{bmatrix} 0 \\ 0 \\ mg - 4 * K_f * \sum \omega_i^2 \\ k_f(\omega_2^2 - \omega_4^2) * l \\ k_f(\omega_3^2 - \omega_1^2) * l \\ -k_m(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dot{\theta}\dot{\psi}(I_{zz} - I_{yy}) \\ \dot{\psi}\dot{\phi}(I_{xx} - I_{yy}) \\ \dot{\phi}\dot{\theta}(I_{yy} - I_{xx}) \end{bmatrix}$$
$$\begin{bmatrix} \vec{a}_x \\ \vec{a}_y \\ \vec{a}_z \\ \vec{\alpha}_x \\ \vec{\alpha}_y \\ \vec{\alpha}_z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ g - 4 * \frac{K_f}{m} * \sum \omega_i^2 \\ \frac{1}{I_{xx}}(k_f(\omega_2^2 - \omega_4^2) * l - \dot{\theta}\dot{\psi}(I_{zz} - I_{yy}) \\ \frac{1}{I_{yy}} k_f(\omega_3^2 - \omega_1^2) - \dot{\psi}\dot{\phi}(I_{xx} - I_{yy}) \\ \frac{1}{I_{zz}}(-k_m(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) - \dot{\phi}\dot{\theta}(I_{yy} - I_{xx}) \end{bmatrix} \tag{23}$$

Therefore, we have now described the equations of motions of the quadcopter drone using its own reference frame coordinates which will come in handy during the testing section.

III. Controller Design and Stability

To accomplish the overall goal of driving the drone to its target, the ePFC paper, or the PFC algorithm in general, utilizes the simple analogy of connecting the drone with the target using a spring, which as a result will pull the drone towards the target; and a compressed spring attached from obstacle to the drone that would like to push away the drone until reaching a safe distance. Similarly, potential field controllers, as shown in Fig. 7, are designed with attractive and repulsive forces, which when summed together generate a force field map for the drone so that it can drive to the target while avoiding obstacles.

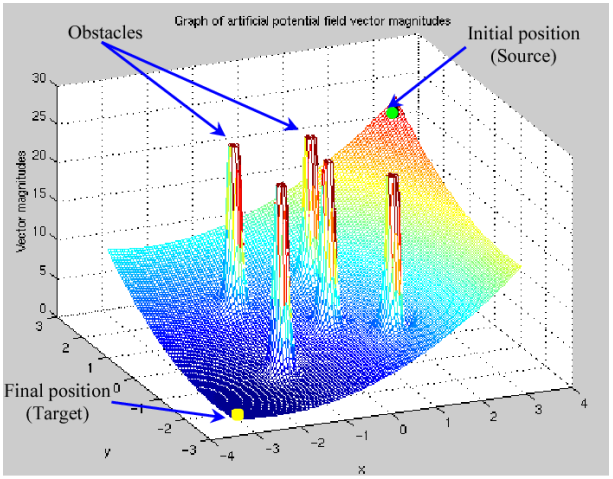

Fig. 1. Graph of artificial field vector magnitudes [14].

Fig. 7. Potential Field Control map with obstacles and targets shown as gradients to lead the drone towards the target [27]

A. Controller Design

We start by defining position vectors for the drone and target location as $\vec{p_D} = [x_D, y_D, z_D]^T$ and $\vec{p_T} = [x_T, y_T, z_T]^T$ respectively. Furthermore, the relative distance between the drone and target will then be

$$\begin{aligned} \vec{p_{dt}} &= \vec{p_d} - \vec{p_t} \\ &= [x_D, y_D, z_D]^T - [x_T, y_T, z_T]^T \\ &= [x_{DT}, y_{DT}, z_{DT}]^T \end{aligned} \tag{24}$$

Potential forces are defined in spatial dimensions and defined by a quadratic function given by,

$$U_{att^1}(\vec{p_{DT}}) = \frac{1}{2}\lambda_1 ||\vec{p_{DT}}||^2 \tag{25}$$

with λ_1 being an empirically determined scaling factor while $||\vec{p_{dt}}||$ is the norm of the relative position vector between drone and target given by,

$$||\vec{p_{DT}}|| = \sqrt{x_{DT}^2 + y_{DT}^2 + z_{DT}^2} \tag{26}$$

In order for the drone to achieve its goal of reaching the target, the source would need to thread downhill and arrive at the global minimum spot as shown in Fig. 7; and this can be accomplished by taking negative gradients of the potential field equation shown below.

$$
\begin{aligned}
&\vec{v_{DT}}^{att^1} = -\nabla U^1_{att}(p_{DT}) \\
&= -\frac{\partial U_{att^1}}{\partial x}\hat{i} - \frac{\partial U_{att^1}}{\partial y}\hat{j} - \frac{\partial U_{att^1}}{\partial z}\hat{k} \\
&= -\nabla(\tfrac{1}{2}\lambda_1 ||\vec{p_{DT}}||^2) \\
&= -\tfrac{1}{2}\nabla(\vec{x_{DT}}^2 + \vec{y_{DT}}^2 + \vec{z_{DT}}^2) \\
&= -\lambda_1(\vec{x_{DT}} + \vec{y_{DT}} + \vec{z_{DT}} \\
&\vec{v_{DT}}^{att^1} = -\lambda_1(\vec{p}_{DT})
\end{aligned}
\tag{27}
$$

Note that $\vec{v_{DT}}^{att^1}$ is the attractive velocity which "pulls" the drone towards the target, hence simple attractive PFC. Since the workspace includes obstacles to avoid as well, the repulsive potential would be the inverse of square of the relative distance, ie square because we're working with Potenial Fields and inverse because the relative proximity maximizes the potential as shown by the spikes in Fig. 7. Thus, the repulsive potential is represented as,

$$
U_{rep^1}(\vec{p_{DO}}) = \frac{1}{2}\eta_1 \frac{1}{||\vec{p_{DO}}||^2} \tag{28}
$$

where η_1 is an emperical scaling factor while $\vec{p_{DO}}$ is the separation distance between the drone and the obstacle in question. The desired repulsive velocity $\vec{v_{DO}}^{rep^1}$, similar to the attraction results, can be obtained by deriving the repulsive potential field as follows:

$$
\begin{aligned}
&\vec{v_{DO}}^{rep^1} = -\nabla U^1_{rep}(p_{DO}) \\
&= -\frac{\partial U_{rep^1}}{\partial x}\hat{i} - \frac{\partial U_{rep^1}}{\partial y}\hat{j} - \frac{\partial U_{rep^1}}{\partial z}\hat{k} \\
&= -\nabla(\tfrac{1}{2}\eta_1 \tfrac{1}{||\vec{p_{DO}}||^2}) \\
&= -\tfrac{1}{2}\nabla(\tfrac{1}{\vec{x_{DO}}^2 + \vec{y_{DO}}^2 + \vec{z_{DO}}^2}) \\
&= \eta_1 \tfrac{\vec{x_{DO}} + \vec{y_{DO}} + \vec{z_{DO}}}{(\vec{x_{DO}}^2 + \vec{y_{DO}}^2 + \vec{z_{DO}}^2)^2} \\
&\vec{v_{DO}}^{rep^1} = \eta_1 \tfrac{\vec{p}_{DO}}{||\vec{p}_{DO}||^4}
\end{aligned}
\tag{29}
$$

Hence, as the drone gets closer to the obstacle, the denominator increases the overall velocity, while increased separation distance reduces the repulsive velocity to zero. Nonetheless, by design, the repulsive velocity can be assigned a cutoff radius from the obstacle, thenceforth distance greater than that, say P^* as shown in Fig. 8, will not feel the effects from the field.

As a result, the repulsive velocity terms can be succinctly shown as

$$
\vec{v}^{rep^1}_{DO} = \begin{cases} \eta_1 \frac{\vec{p}_{DO}}{||\vec{p}_{DO}||^4}, & ||\vec{p_{DO}}|| \leq P^* \\ 0, & ||\vec{p_{DO}}|| > P^* \end{cases} \tag{30}
$$

Finally, the complete traditional PFC gradient equation for tracking a target and avoiding n obstacles would be sum of 27 and 30 that comes out to be

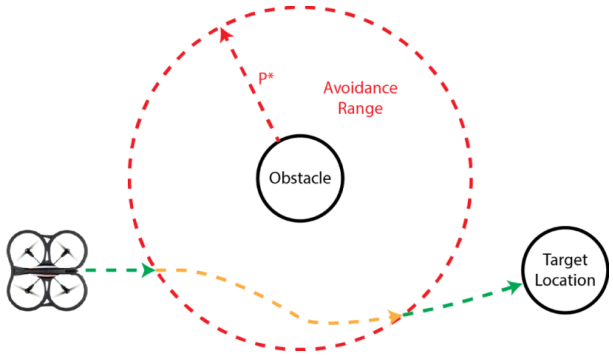

Fig. 8. Quadcopter drone changing trajectory as it enters the Obstacle proximity circle due to repulsive velocity push. P^* is the fixed radius cut of range chosen to mark region where repulsive potential field will begin effect.

[28]

$$
\vec{v}_D^{PFC} = \begin{cases} -\lambda_1(\vec{p}_{DT}) + \sum_{i=0}^{n} \eta_1 \frac{\vec{p}_{DO}}{||\vec{p}_{DO}||^4}, & ||\vec{p}_{DO}|| \leq P^* \\ -\lambda_1(\vec{p}_{DT}), & ||\vec{p}_{DO}|| > P^* \end{cases} \tag{31}
$$

which is a demonstrates to be suitable controller for ground robots, but very poor in performance for agile aerial systems as seen in later simulation section, which the ePFC paper proves to have solved.

B. Extended Potential Field Controller

As stated above, unlike ground robots, aerial robots are always making adjustments to either hover in place, move in a steady path or make sudden changes in trajectory due to their inherent instability. As a result, to provide more instant responses to changes in the environment, the relative velocity between drone and target is taken as an input, thus updating the traditional PFC to work with velocities instead as well as track dynamic targets. Hence, the quadratic attractive potential function is defined as,

$$
U_{att^2}(\vec{v_{DT}}) = \frac{1}{2}\lambda_2 ||\vec{v_{DT}}||^2 \tag{32}
$$

with λ_2 denoting an empirically computed positive scale factor, while $||\vec{v_{DT}}||$ is the norm (magnitude) of the relative velocity between the drone and target given by

$$
||\vec{v}_{DT}|| = \sqrt{\dot{x_{DT}}^2 + \dot{y_{DT}}^2 + \dot{z_{DT}}^2} \tag{33}
$$

The next task would be to calculate the negative gradient of the attractive potential so that the velocity of the drone and the target match. The attractive velocity is then,

$$
\begin{aligned}
&\vec{v_{DT}}^{att^2} = -\nabla U^2_{att}(v_{DT}) \\
&= -\frac{\partial U_{att^2}}{\partial \dot{x}}\hat{i} - \frac{\partial U_{att^2}}{\partial \dot{y}}\hat{j} - \frac{\partial U_{att^2}}{\partial \dot{z}}\hat{k} \\
&= -\nabla(\tfrac{1}{2}\lambda_2 ||\vec{v_{DT}}||^2) \\
&= -\tfrac{1}{2}\nabla(\dot{\vec{x}}^2_{DT} + \dot{\vec{y}}^2_{DT} + \dot{\vec{z}}^2_{DT}) \\
&= -\lambda_2(\dot{\vec{x}}_{DT} + \dot{\vec{y}}_{DT} + \dot{\vec{z}}_{DT}) \\
&\vec{v_{DT}}^{att^2} = -\lambda_2(\vec{v}_{DT})
\end{aligned}
\tag{34}
$$

On the other hand, since we want to avoid obstacles in the space colliding with drone, we take the inverse quadratic to exert a repulsive field when the drone approaches a given obstacle, similar to the traditional PFC given by,

$$U_{rep^2}(\vec{v}_{DO}) = \frac{1}{2}\eta_2 \frac{1}{||\vec{v}_{DO}||^2} \tag{35}$$

with a positive scaling factor of η_2 and magnitude of relative velocity between drone and obstacle denoted by $||\vec{v}_{DO}||$. Then, the negative gradient of the repulsive potential, ie the repulsive velocity, can be determined as follows,

$$\begin{aligned}
\vec{v}_{DO}^{rep^2} &= -\nabla U_{rep}^2(v_{DO}) \\
&= -\frac{\partial U_{rep^2}}{\partial \dot{x}}\hat{i} - \frac{\partial U_{rep^2}}{\partial \dot{y}}\hat{j} - \frac{\partial U_{rep^2}}{\partial \dot{z}}\hat{k} \\
&= -\nabla(\tfrac{1}{2}\eta_2 \tfrac{1}{||\vec{v}_{DO}||^2}) \\
&= -\tfrac{1}{2}\nabla(\tfrac{1}{\dot{x}_{DO}^2+\dot{y}_{DO}^2+\dot{z}_{DO}^2}) \\
&= \eta_2 \tfrac{\dot{x}_{DT}+\dot{y}_{DT}+\dot{z}_{DT}}{(\dot{x}_{DT}^2+\dot{y}_{DT}^2+\dot{z}_{DT}^2)^2} \\
\vec{v}_{DO}^{rep^2} &= \eta_2 \tfrac{\vec{v}_{DO}}{||v\vec{D}O||^4}
\end{aligned} \tag{36}$$

Because this analysis is done with the assumption that both drone and obstacle are in motion, when computing repulsive velocity for stationary obstacles, v_O is zero; hence, to include both cases, the repulsive velocity can be expressed as,

$$\vec{v}_{DO}^{rep^2} = \begin{cases} \eta_2 \frac{\vec{v}_{DO}}{||\vec{v}_{DO}||^4}, & ||\vec{v}_{DO}|| \neq 0 \\ 0, & ||\vec{v}_{DO}|| = 0 \end{cases} \tag{37}$$

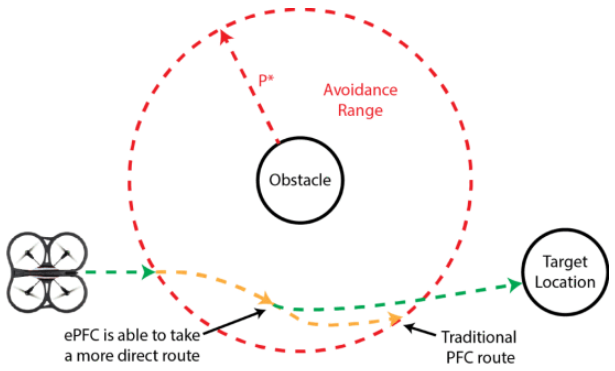

Fig. 9. Quadcopter drone changing trajectory as it enters the Obstacle proximity zone due to repulsive velocity push but quickly adjusting route and ignoring push velocity effect once quadcopter initiates avoidance [28]

At the end, the relative distance parameter is reevaluated so that the drone's holistic performance is captured. That is, the drone and target could have their relative velocities approximated, and yet still travel with a constant separation distance which will prevent the drone from reaching its goal. On the other hand, if relative position is only considered as in the traditional PFC case, there will be a lag in the drone's reaction time. For instance, if the drone enters the proximity region of the obstacle but quickly begin to travel away from it while still being in the zone, traditional PFC would still apply repulsive force although the drone is already in a receding motion, which will delay the quadcopter from getting back to its optimal trajectory, unlike the ePFC shown in Fig. 9 resolving issue. Similarly, if the drone is outside the obstacle's proximity zone, but drives towards it, evasive action needs to be taken. Thus, the repulsive velocity can be summarized based on the drone to obstacle relative velocity as,

$$\vec{v}_{DO}^{rep^3} = \begin{cases} -\eta_3 ||\dot{\vec{p}}_{DO}|| \frac{\vec{p}_{DO}}{||\vec{p}_{DO}||}, & ||\dot{\vec{p}}_{DO}|| < 0 \\ 0, & ||\dot{\vec{p}}_{DO}|| \geq 0 \end{cases} \tag{38}$$

where η_3 is a positive scaling factor, and the reduction in separation distance between obstacle and drone is countered with an opposing gradient of the potential.

Therefore, the full expression of the extended Potential Field Control will be a sum of the traditional PFC, [31], and the attractive, [34] , as well as the two repulsive velocities (gradients) between the quadcopter drone and respective target, [37], and obstacle [38] objects within the environment. Hence, the full ePFC form, for n obstacles is

$$\begin{aligned}
\vec{v}_d^{ePFC} &= \vec{v}^{PFC} - \lambda_2(\vec{v}_{DT}) + \\
&\sum_{i=0}^{n} \eta_2 \tfrac{\vec{v}_{DO_i}}{||\vec{v}_{DO_i}||^4} - \sum_{i=0}^{n} \eta_3 ||\dot{\vec{p}}_{DO}|| \tfrac{\vec{p}_{DO}}{||\vec{p}_{DO}||^4}
\end{aligned} \tag{39}$$

where the case for the above expression is that the obstacle is moving with some velocity v_{O_i} with drone approaching it. This form, however, would need to be transformed from the fixed world frame to the drone's reference frame - as discussed in section *II*, so that we can retrieve the desired yaw angle (as pitch and yaw have been neglected in assumptions from previous sections) and pass yaw commands to the drone to establish rotational control as drone navigates in environment, obtained by

$$\begin{aligned}
\vec{v}_{d_{body}}^{ePFC} &= \vec{v}^{ePFC} * R_z \\
\vec{v}_{d_{body}}^{ePFC} &= \vec{v}^{ePFC} \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{40}$$

with ψ being the yaw angle of the drone with respect to its own z axis.

At this point, we have derived a controller that takes in both relative distance and velocity from a quadcopter and surrounding objects, as well as compute yaw, that aims at driving drone to goal location without bumping into any other unwanted obstacles or barriers. We are now left with determining if controller is stable

C. Stability Analysis

The authors of the ePFC paper have chosen to use Lyapunov theory to analyze convergence of the controller they have proposed. Because the paper uses convex potential field functions

to define attraction and repulsion, while also seeking for a negative gradient to attain global minimum, they have chosen Lyapunov theory. Hence, considering attractive potential as a positive definite Lyapunov function, we get

$$L = U_{att} = \frac{1}{2}\lambda_1||\vec{p}_{DT}||^2 + \frac{1}{2}\lambda_2||\vec{v}_{DT}||^2 \tag{41}$$

In order to obtain the gradient (directional derivatives) for the positive definite potentials (expressed here as vector fields), the Lie Derivative is taken to evaluate the change of the vector field with respect to other fields (here $\vec{p}_{DT}$ and $\vec{v}_{DT}$ yielding

$$\begin{aligned} L^* &= \frac{\partial L}{\partial \vec{p}_{DT}}\frac{\partial}{\partial t}\vec{p}_{DT} + \frac{\partial L}{\partial \vec{p}_{DT}}\frac{\partial}{\partial t}\vec{v}_{DT} \\ &= \frac{\partial L}{\partial \vec{p}_{DT}}\vec{v}_{DT} + \frac{\partial L}{\partial \vec{v}_{DT}}\vec{a}_{DT} \\ L^* &= \lambda_1||\vec{p}_{DT}||\vec{v}_{DT} + \lambda_2||\vec{v}_{DT}||\vec{a}_{DT} \end{aligned} \tag{42}$$

where a_{DT} is the relative acceleration between the drone and target, and can be computed by taking derivative of the relative velocity between the drone and the target 34 as follows,

$$\begin{aligned} \vec{a}_{DT} &= \dot{\vec{v}}_{dt} \\ &= \tfrac{d}{dt}(-\tfrac{1}{2}\lambda_2\nabla(\dot{\vec{x}}^2_{DT} + \dot{\vec{y}}^2_{DT} + \dot{\vec{z}}^2_{DT})) \\ &= \tfrac{d}{dt}(-\lambda_2(\dot{\vec{x}}_{DT} + \dot{\vec{y}}_{DT} + \dot{\vec{z}}_{DT})) \\ &= \tfrac{d}{dt}(-\lambda_2||\vec{v}_{DT}||) \\ \vec{a}_{DT} &= -\lambda_2||\tfrac{d}{dt}\vec{v}_{DT}|| \end{aligned} \tag{43}$$

Hence, since we know that the derivative of the relative distance between the drone and the target (input to the PFC term of the Lyapunov function) is the attractive velocity term (seen in [27] as $\vec{v}_{DT} = -\nabla(\frac{1}{2}\lambda_1 * ||\vec{p}_{DT}||^2)$ that simplifies to $\vec{v}_{DT} = -\lambda||\vec{p}_{DT}||$), then making these substitutions in 42 yields,

$$\begin{aligned} L^* &= \lambda_1||\vec{p}_{DT}||\vec{v}_{DT} + \lambda_2||\vec{v}_{DT}||\vec{a}_{DT} \\ L^* &= \lambda_1||\vec{p}_{DT}||(-\lambda_1||\vec{p}_{DT}) + \lambda_2||\vec{v}_{DT}|| * (-\lambda_2||\tfrac{d}{dt}\vec{v}_{DT}||)) \\ L^* &= -[\lambda_1^2||\vec{p}_{DT}||^2 + \lambda_2^2||\vec{v}_{DT}|| * ||\tfrac{d}{dt}\vec{v}_{DT}||] \end{aligned} \tag{44}$$

The Lie derivative, as a result, shows that it will always be negative, since all of it's terms are either positive scaling factors or norms which compute to give positive numbers. Therefore, the paper proves, as we have confirmed here again, that the proposed extended PFC controller is stable and capable enough to track then reach at a target, either stationary or moving.

IV. Simulation

A. MATLAB Setup

We used a MATLAB Simulink environment to perform the simulation. The AR Drone Simulink Development Kit v1.1 Add-In for MATLAB [29] was used to obtain the Simulink model of the AR Drone's state space representation. We then created our custom PFC/EPFC controller model along with our Target and Obstacle logic. The Simulink model is as shown in Fig. 10.

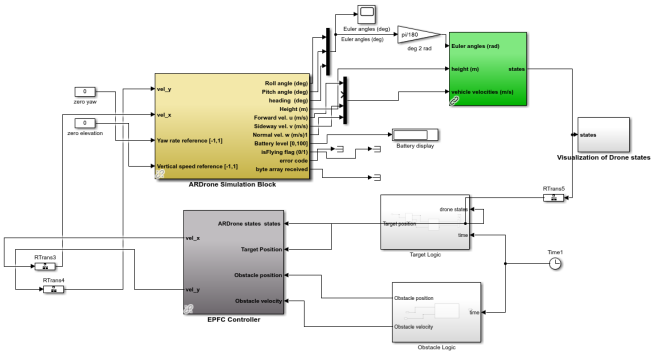

Fig. 10. Simulink model of AR Drone along with ePFC controller

To perform the simulation, first a MATLAB script setupEPFC.m is run, which launches our Simulink model. Then, we press the Run button to run the simulation. Various scopes are available to visualise the x,y,z positions of the drone during the simulation. Additionally, an XY graph is available to visualise the X-Y pose of the drone in time. This scope is helpful to visualise the robot's trajectory.

B. MATLAB Simulation Results

We conducted the simulation for the same waypoint and obstacle configuration as in the original paper. Table I shows the waypoint positions and Table II shows the obstacle position for the simulation environment. After numerous testing interations, $\lambda 1$ = 0.2 and $\eta 1$ = 0.03 were used for the PFC controller. While, for the ePFC controller, we used $\lambda 2$ = 0.05, $\eta 2$ = 0.01 and $\eta 3$ = 2.5.

TABLE I
Simultaion Waypoints

Waypoint	x (m)	y (m)
1	2.5	-1.0
2	2.5	1.0
3	-2.5	1.0
4	-2.5	-1.0
5	2.5	-1.0

TABLE II
Simulation Obstacles

Obstacle	x (m)	y (m)
1	1.0	1.0

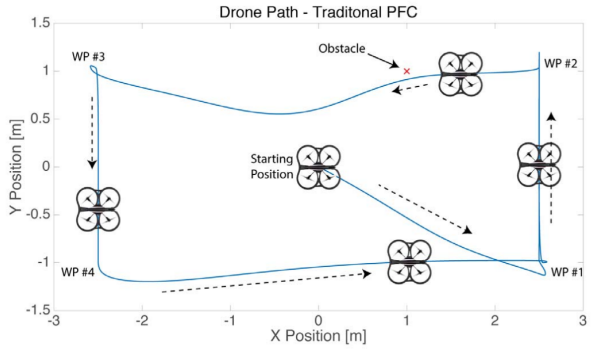

Fig. 11. Simulation of the PFC controller in the Original paper

By comparing Fig. 11 and Fig. 12 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. Also, looking at Fig. 13 and Fig. 14, 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 in Fig. 12 and the ePFC in Fig. 14, we can see that clearly ePFC outperforms PFC with less overshoot and smooth trajectory amid the obstacle's presence.

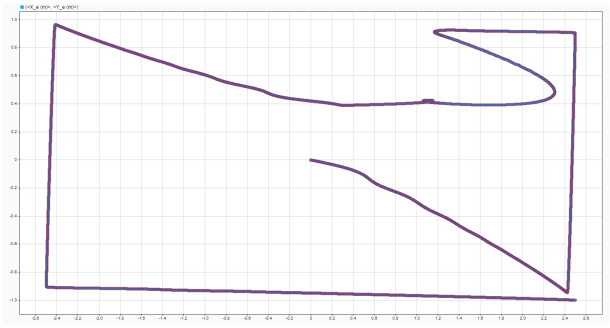

Fig. 12. Our simulation of PFC controller

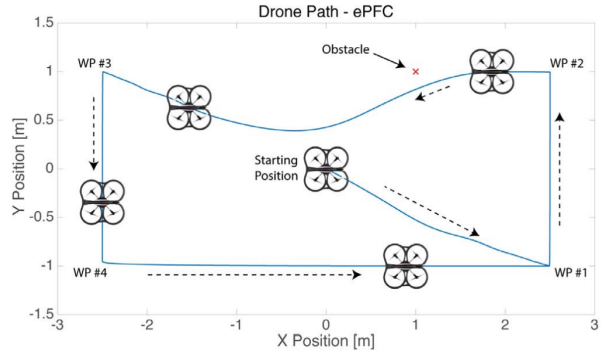

Fig. 13. Simulation of the PFC controller in the Original paper

V. Experimental Setup

Since we did not have the access to the ARDrone hardware, we performed our experiments in a physics simulator. We used the Gazebo simulator in conjunction with the Robot Operating System (ROS) framework. We specifically used ROS Noetic. We performed our experiments in a Ubuntu 20.04 system.

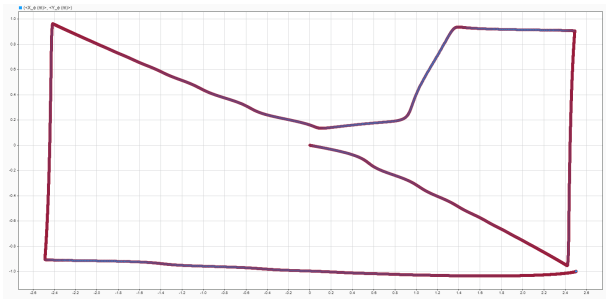

Fig. 14. Our simulation of ePFC controller

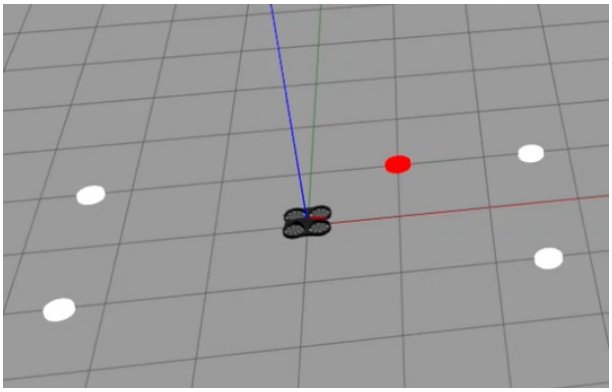

Fig. 15. Gazebo world with waypoints indicated with white and obstacles indicated with red

We conducted six experiments and for each experiment we have a different combination of waypoints and obstacles. Hence, we created six gazebo worlds such as Fig. 15 with white tablets indicating the position of waypoints/targets and red tablets indicating the position of obstacles. We employed the sjtu-drone ROS package [30] to effectively simulate the AR Drone in the Gazebo worlds that we created. Additionally, a forked ROS package [31] of the sjtu-drone was helpful in understanding controller implementation on the AR Drone in Gazebo simulation.

VI. Experimental Results

We conducted a total of six experiments as done in the original paper. Each experiment investigates a different aspect of the developed controller.

A. Experiment 1

The first experiment is setup as identical to the Simulink simulation setup with four targets around the drone as shown in Table. III and one obstacle in it's way as observed in Table. IV.

First, we simulate the traditional PFC with $\lambda 1 = 1.4$ and $\eta 1 = 0.032$. The resulting trajectory can be observed in Fig. 16.

The extended PFC is simulated with $\lambda 1 = 1.4$, $\eta 1 = 0.035$ and $\lambda 2 = 0.28$. The resulting trajectory can be observed in Fig. 17.

TABLE III
EXPERIMENT I WAYPOINTS

Waypoint	x (m)	y (m)
1	2.5	-1.0
2	2.5	1.0
3	-2.5	1.0
4	-2.5	-1.0
5	2.5	-1.0

TABLE IV
EXPERIMENT I OBSTACLES

Obstacle	x (m)	y (m)
1	1.0	1.0

Comparing the trajectories of the drone using both the controllers, we can clearly see that the ePFC Fig. 17 performs better than the PFC Fig. 16 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.

B. Experiment 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. For simplicity, we define the waypoints that make the drone move in a straight line. Kindly note that we have shifted the waypoints by a distance of 3m in the x-direction so that our drone starts at origin. The waypoints are as shown in Table. V and obstacles as shown in Table. VI.

TABLE V
EXPERIMENT II WAYPOINTS

Waypoint	x (m)	y (m)
1	6.0	0.0
2	0.0	0.0

TABLE VI
EXPERIMENT II OBSTACLES

Obstacle	x (m)	y (m)
1	1.25	0.0
2	2.5	0.5
3	3.0	0.0
4	4.25	0.5
5	4.75	-0.5

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 Fig. 19 and the ePFC Fig. 20. 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 $\lambda 2 = 0.28$. 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. Kindly note that the Fig. 20 may be misleading at the point (3,0). Here, it seems like the drone has collided with the

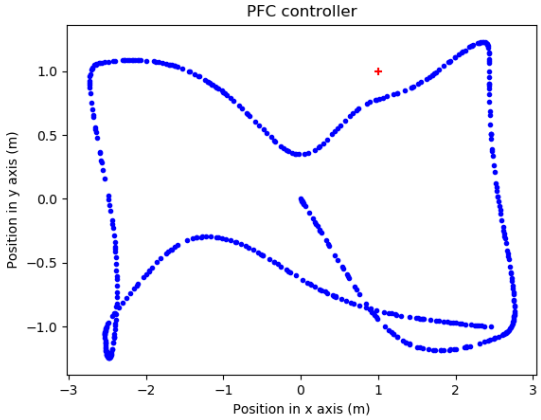

Fig. 16. AR Drone trajectory using PFC controller

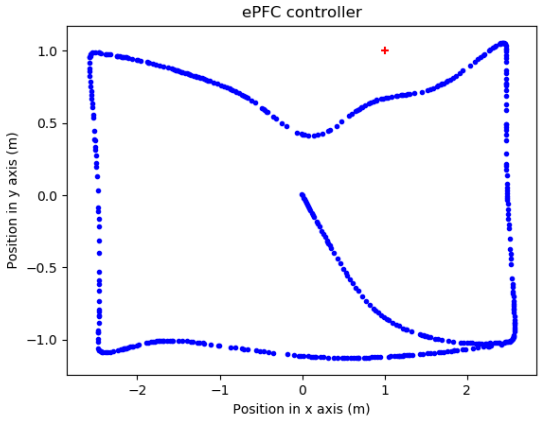

Fig. 17. AR Drone trajectory using ePFC controller

obstacle. But, that is not the case. If the plot is looked closely, it may be clear that the drone indeed avoids the obstacle at (3,0).

C. 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. The position of the waypoint can be observed in Table. VII.

TABLE VII
EXPERIMENT III WAYPOINTS

Waypoint	x (m)	y (m)
1	4.25	0.0

In our experiment we found that the drone takes 5.5 seconds (t=545 to t=550.5) as shown in Fig. 22 to reach the target. This figure is quite close to the one in the original paper (5 seconds) as shown in Fig. 21.

D. Experiment 4

In this experiment, we are interested in finding out the change in relative distance between the drone and target ans

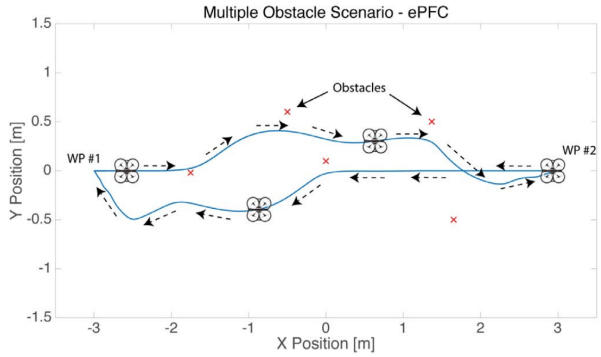

Fig. 18. AR Drone trajectory with multiple obstacles in the original paper

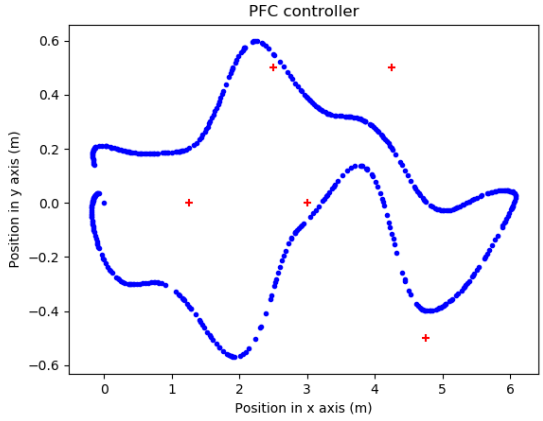

Fig. 19. AR Drone trajectory with multiple obstacles using PFC controller

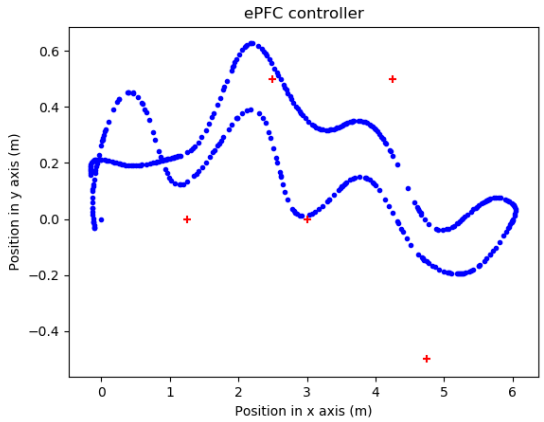

Fig. 20. AR Drone trajectory with multiple obstacles using ePFC controller

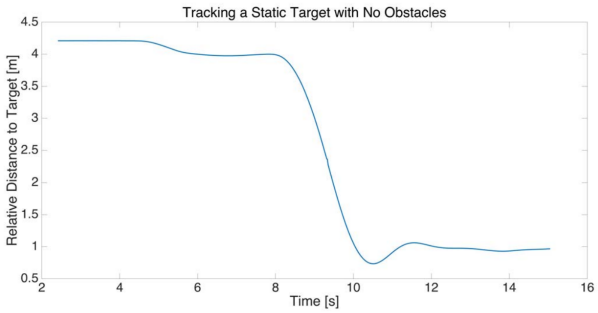

Fig. 21. Result of tracking target with no obstacles in original paper

well as between the drone and the obstacle. Hence, we create a setup where we have an obstacle Table. IX in-between the drone and the waypoint Table. VIII.

TABLE VIII
EXPERIMENT IV WAYPOINTS

Waypoint	x (m)	y (m)
1	5.25	0.0

TABLE IX
EXPERIMENT IV OBSTACLES

Obstacle	x (m)	y (m)
1	2.5	0.0

Comparing the results obtained in the original paper Fig. 23 with the one we obtained Fig. 24, We see that the nature of the graph obtained in our experiment is similar to the one in the original paper.

E. Experiment 5

In this experiment we test the controller on a setup similar to that of the simulation but with closer waypoints Table. X and an obstacle Table. XI. We notice the drone's trajectory in the original paper Fig. 25 and observe that the trajectory obtained from our experiment Fig. 26 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.

TABLE X
EXPERIMENT V WAYPOINTS

Waypoint	x (m)	y (m)
1	-1.5	-0.5
2	1.5	-0.5
3	1.5	0.5
4	-1.5	0.5
5	-1.5	-0.5

TABLE XI
EXPERIMENT V OBSTACLES

Obstacle	x (m)	y (m)
1	0.0	0.38

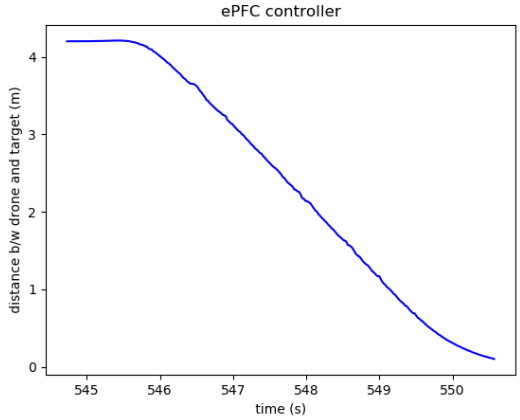

Fig. 22. Our result of tracking target with no obstacles

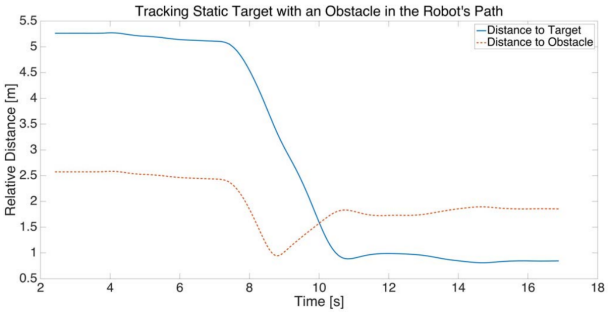

Fig. 23. Result of tracking target with an obstacle in-between in original paper

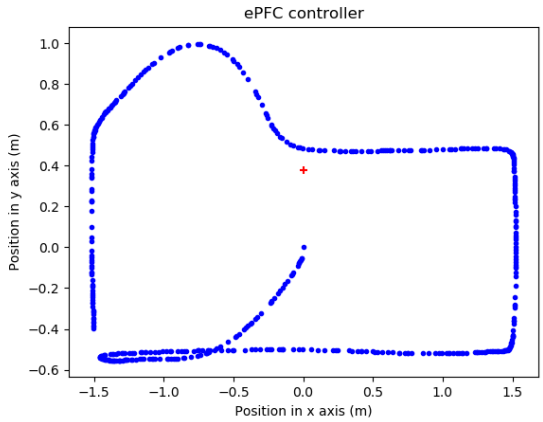

Fig. 26. AR Drone trajectory using ePFC in our Gazebo simulation

F. 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 and also is offset the y-direction Table. XII. We are interested in observing the tracking error in the x-direction.

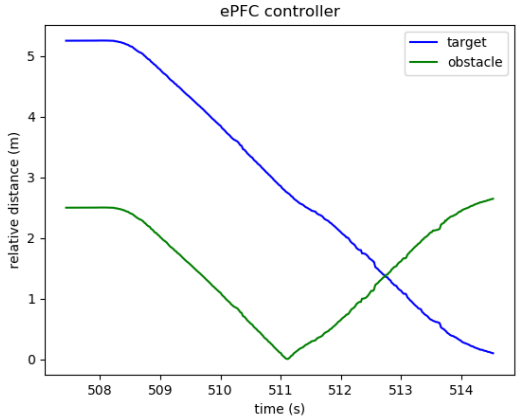

Fig. 24. Our result of tracking target with an obstacle in-between

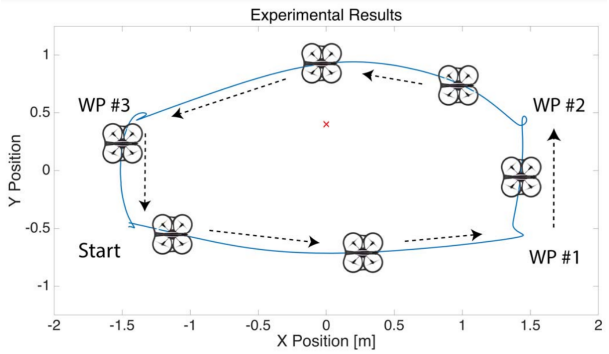

Fig. 25. AR Drone trajectory using ePFC in the original paper

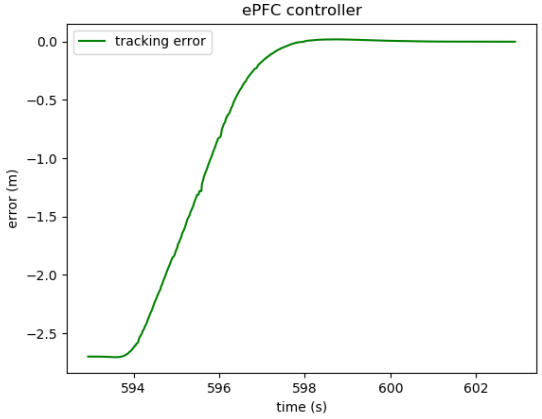

Fig. 27. Error in waypoint tracking in our experiment

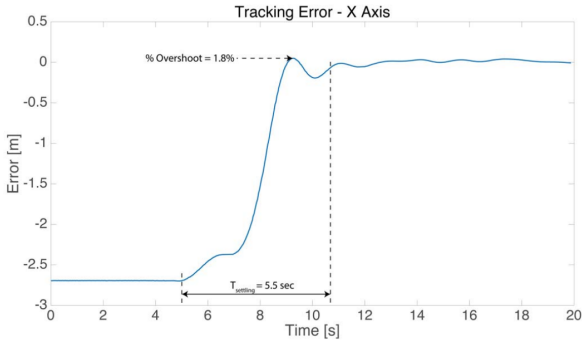

Fig. 28. Error in waypoint tracking in original paper

TABLE XII
EXPERIMENT VI WAYPOINTS

Waypoint	x (m)	y (m)
1	2.7	1.0

We observe that the settling time for the drone in our case is 6s (t=594 to t=600) Fig. 27 which is close to the one in the original paper of 5.5s Fig. 28. 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.

VII. CONCLUSION

In 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.

REFERENCES

[1] R. Barua and S. Datta, "Modernization of robotics application in 21 st century: A review," *Journal of Mechanical Robotics*, vol. 26, no. 36, October 2020.

[2] K. K. M. I. Bernard, M. and A. Ollero, "Autonomous transportation and deployment with aerial robots for search and rescue missions," *Journal ofField Robotics*, no. 28, pp. 914–931, 2011.

[3] M. Hossain, "Autonomous delivery robots: A literature review," *Engineering Management Review*.

[4] S. Young and A. Kott, "A survey of research on control of teams of small robots in military operations," *CoRR*, vol. abs/1606.01288, 2016. [Online]. Available: http://arxiv.org/abs/1606.01288

[5] D. González-Aguilera and P. Rodríguez-Gonzálvez, "Drones—an open access journal," *Drones*, vol. 1, p. 1, 01 2017.

[6] M. Chapman. (2018, February) Firefighter using a drone/uav stock photo. [Online]. Available: https://www.istockphoto.com/photo/firefighter-using-a-drone-uav-gm899442412-248190530?

[7] A. Viswanathan, B. R. Pires, and D. Huber, "Vision based robot localization by ground to satellite matching in gps-denied situations," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 192–198.

[8] M. Shaohua, X. Jinwu, and L. Zhangping, "Navigation of micro aerial vehicle in unknown environments," in *2013 25th Chinese Control and Decision Conference (CCDC)*, 2013, pp. 322–327.

[9] S. Grzonka, G. Grisetti, and W. Burgard, "A fully autonomous indoor quadrotor," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 90–100, 2012.

[10] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained mav," 06 2011, pp. 20 – 25.

[11] I. Sa and P. Corke, "System identification, estimation and control for a cost effective open-source quadcopter," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2202–2209, 05 2012.

[12] A. G. Kendall, N. N. Salvapantula, and K. A. Stol, "On-board object tracking control of a quadcopter with monocular vision," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014, pp. 404–411.

[13] S. Shen, N. Michael, and V. Kumar, "Autonomous indoor 3d exploration with a micro-aerial vehicle," in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 9–15.

[14] J.-E. Gomez-Balderas, P. Castillo Garcia, J. Guerrero, and R. Lozano, "Vision based tracking for a quadrotor using vanishing points," *Journal of Intelligent and Robotic Systems*, vol. 65, pp. 361–371, 07 2012.

[15] L. Meier, P. Tanskanen, L. Heng, G. Lee, F. Fraundorfer, and M. Pollefeys, "Pixhawk: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Autonomous Robots*, vol. 33, 08 2012.

[16] Y. S. Alqudsi, A. H. Kassem, and G. M. El-Bayoumi, "Trajectory generation and optimization algorithm for autonomous aerial robots," in *2021 1st International Conference on Emerging Smart Technologies and Applications (eSmarTA)*, 2021, pp. 1–8.

[17] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[18] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The grasp multiple micro-uav testbed," *IEEE Robotics Automation Magazine*, vol. 17, no. 3, pp. 56–65, 2010.

[19] A. Dogan, "Probabilistic approach in path planning for uavs," in *Proceedings of the 2003 IEEE International Symposium on Intelligent Control*, 2003, pp. 608–613.

[20] Y. Kuwata, T. Schouwenaars, A. Richards, and J. How, "Robust constrained receding horizon control for trajectory planning," 08 2005.

[21] A. Neto, D. Macharet, and M. Campos, "Feasible rrt-based path planning using seventh order bézier curves," 11 2010, pp. 1445 – 1450.

[22] K. Kim and C. S. Hong, "Optimal task-uav-edge matching for computation offloading in uav assisted mobile edge computing," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2019, pp. 1–4.

[23] A. Noordin, A. Basri, and Z. Mohamed, "Sensor fusion for attitude estimation and pid control of quadrotor uav," *International Journal of Electrical and Electronic Engineering Telecommunications.*, vol. 7, pp. 183–189, 10 2018.

[24] C. Troiani, A. Martinelli, C. Laugier, and D. Scaramuzza, "2-point-based outlier rejection for camera-imu systems with applications to micro aerial vehicles," 05 2014.

[25] Semath.info. (2022) How to derive rotation matrix by euler angles. [Online]. Available: https://semath.info/img/euler_angle_fig_00.jpg

[26] ——. (2022) How to derive rotation matrix by euler angles. [Online]. Available: https://semath.info/img/euler_angle_fig_01.jpg

[27] T. Stirling, J. Roberts, J.-C. Zufferey, and D. Floreano, "Indoor navigation with a swarm of flying robots," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4641–4647, 11 2012.

[28] A. C. Woods and H. M. La, "A novel potential field controller for use on aerial robots," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 4, pp. 665–676, 2019.

[29] D. E. Sanabria. (2023) Ar drone simulink development kit v1.1. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/43719-ar-drone-simulink-development-kit-v1-1

[30] tahsinkose. (2021) sjtu drone. [Online]. Available: https://github.com/tahsinkose/sjtu-drone
[31] A. Malapaka. (2021) Ardrone potential field controller. [Online]. Available: https://github.com/adarshmalapaka/ardrone-potential-field-controller