A Novel Potential Field Controller for Use on Aerial Robots

Alexander C. Woods and Hung M. La, *Senior Member, IEEE*

***Abstract*—Unmanned aerial vehicles, commonly known as drones, have many potential uses in real-world applications. Drones require advanced planning and navigation algorithms to enable them to safely move through and interact with the world around them. This paper presents an extended potential field controller (ePFC) which enables an aerial robot, or drone, to safely track a dynamic target location while simultaneously avoiding any obstacles in its path. The ePFC outperforms a traditional potential field controller with smoother tracking paths and shorter settling times. The proposed ePFC's stability is evaluated by Lyapunov approach, and its performance is simulated in a MATLAB environment. Finally, the controller is implemented on an experimental platform in a laboratory environment which demonstrates the effectiveness of the controller.**

***Index Terms*—Aerial robots, drones, potential field control, unmanned autonomous systems (UAS), unmanned autonomous vehicles (UAV).**

I. Introduction

THIS paper focuses on dynamic target tracking and obstacle avoidance on a quadcopter drone such as the one shown in Fig. 1. Recent advances in the field of unmanned autonomous systems (UASs) have drastically increased the potential uses of both unmanned ground vehicles and unmanned aerial vehicles (UAVs) [1], [2]. UAS can be utilized in situations which may be hazardous to human operators in ground vehicles or pilots in traditional aircraft, such as assisting wild land fire fighters [3]–[7], search and rescue operations in unsafe conditions or locations [8]–[12], and disaster relief efforts [13]–[15]. Additionally, UAS can be used in repetitive or tedious work where a human operator may lose focus such as infrastructure inspection [16]–[18], agricultural inspections [19], [20], and environmental sensing [21], [22]. Specifically, quadcopter systems are desirable because they can perform very agile maneuvers which gives them an advantage over fixed-wing platforms in confined environments.

Fig. 1. Low-cost, commercially available quadcopter drone is used as the experimental platform for testing and demonstrating the effectiveness of the proposed control method.

Although the field of UAS has grown rapidly, it is still hindered by many problems which limit their use in real-world applications. The challenge of localizing in GPS-denied environments has been approached by a multitude of research groups across the world, and there are several methods which have been to address this. One of several promising onboard sensing methods is light detection and ranging (LIDAR). One group employed a reflexive algorithm in combination with an LIDAR sensor for simulating navigation through an unknown environment [23]. Another group developed a multilevel simultaneous localization and mapping algorithm which utilized LIDAR as its primary sensing method [24]. Other groups used LIDAR on autonomous vehicles for control and multifloor navigation [25], [26].

Another major area of research for localization in GPS-denied environments is computer vision. One group used a single camera, looking at an object of known size to determine the drone's location [27]. Another group utilized a combination of LIDAR and a Microsoft Kinect sensor to explore an unknown environment [28]. Several other groups successfully used unique variations of computer vision methods as a means of localization [29], [30], and it is proving to be a very promising method of operating in GPS-denied environments.

In addition to advanced sensing capabilities, UAS also require planning and navigation algorithms to safely move through and interact with the world around them. Trajectory generation for aerial robots has been accomplished through methods such as minimizing snap, the second derivative of acceleration [31], [32]. Given keyframes consisting of a position in space coupled with a yaw angle, this method is able to generate very smooth, optimal trajectories. Other

Manuscript received August 25, 2016; revised March 25, 2017; accepted May 6, 2017. Date of publication May 23, 2017; date of current version March 15, 2019. This work was supported in part by the National Science Foundation under Grant NSF-NRI 1426828, in part by the National Aeronautics and Space Administration through the Nevada NASA Research Infrastructure Development Seed Grant under Grant NNX15AI02H, and in part by the INSPIRE University Transportation Center through the U.S. Department of Transportation Office of the Assistant Secretary for Research and Technology under Grant 69A3551747126. This paper was recommended by Associate Editor S. Nahavandi. *(Corresponding author: Hung M. La.)*

The authors are with the Advanced Robotics and Automation Laboratory, Department of Computer Science and Engineering, University of Nevada at Reno, Reno, NV 89557-0312 USA (e-mail: hla@unr.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2017.2702701

groups successfully applied methods utilizing Voronoi diagrams [27], [33], receding horizons in relatively unrestricted environments [34], high-order parametric curves [35], and 3-D interpolation [36].

However, many platforms do not have the luxury of a very powerful processor and solving complex algorithms cannot practically be performed by an off-board computer. Therefore, the contribution of this paper is to propose an ePFC as a navigation method which is computationally inexpensive, can react quickly to the environment, and which can be deployed onboard any platform with adequate sensing capabilities. The ePFC expands on the basic capabilities of a traditional potential field controller (PFC), which operates only on x and y relative distances, and goes further to intelligently incorporate relative velocity as well. The developed controller's stability is evaluated, and its performance is both simulated and demonstrated experimentally.

The remainder of this paper is organized as follows. Section II presents a system model for the quadcopter system dynamics. Section III provides a brief background on potential field methods, discusses the design of the controller, and demonstrates the stability of the system using a Lyapunov approach. Section IV presents simulation results of the controller implemented in a MATLAB environment. Section V discusses the experimental quadcopter platform, the testing environment, experimental results, and an evaluation of the controller's performance. Finally, Section VI provides a brief conclusion, with recommendations for future work.

II. System Model

This section presents the set of differential equations represent the quadcopter system dynamics. Developing a mathematical model of a system is a fundamental step in any controller design and develops a deeper understanding of the system in question. Once the mathematical system and initial controller design are complete, the combined system can be simulated and tested in an experimental setting. A brief background on Newtonian reference frames is provided, as well as the method used to transform between various reference frames and describe the motion of a rigid body. Finally, the equations of motion are derived using the Newton–Euler method.

A. Reference Frames

To begin, it is important to introduce a set of reference frames which allow representation the position and orientation of a rigid body in space. These reference frames are defined by a linearly independent set of vectors which span the dimensions of the frame. The position and orientation of the rigid body at any instant in time is represented by a particle at its center of mass and is described relative to a Cartesian reference frame in Euclidean space, $E \in \mathbb{R}^3$, which is fixed on earth at a known location as shown in Fig. 2. It is assumed that this reference frame is nonaccelerating, and is therefore inertial. Additionally, the curvature of the earth is considered negligible for the scope of this paper. The axes of $E \in \mathbb{R}^3$ are described by the set of orthogonal unit vectors $(\hat{e}_x, \hat{e}_y, \hat{e}_z) \in \mathbb{R}^3$, where

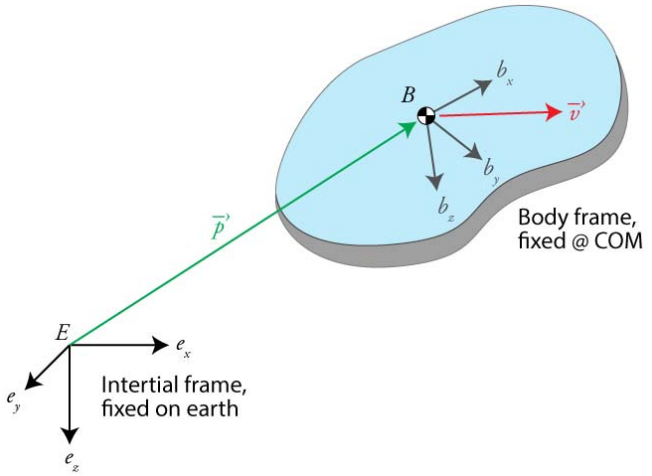

Fig. 2. Inertial reference frame, E, is fixed on earth, and an accelerating body reference frame, B, is fixed at the rigid body's center of mass. The position and velocity of the rigid body are designated as $\vec{p}$ and $\vec{v}$, respectively. Euler rotation matrices may be used to transform between frames E and B for a given orientation.

$\hat{e}_x$ points north, $\hat{e}_y$ points east, and $\hat{e}_z$ points toward the center of the earth.

To simplify the derivation of the equations of motion, an additional reference frame, $B \in \mathbb{R}^3$, is defined which is attached to the particle at the center of mass of the rigid body. This reference frame is referred to as the body frame and is represented by the set of orthogonal unit vectors $(\hat{b}_x, \hat{b}_y, \hat{b}_z) \in \mathbb{R}^3$, where $\hat{b}_x$ points forward, $\hat{b}_y$ points right, and $\hat{b}_z$ points downward, perpendicular to the body. To describe the orientation of the body, name rotation about the $\hat{b}_x$ to be roll, $\vec{\phi}$, rotation about $\hat{b}_y$ to be pitch, $\vec{\theta}$, and rotation about $\hat{b}_z$ to be yaw, $\vec{\psi}$. It is important to note that the body frame is noninertial and does experience acceleration.

B. Euler Transformations

In order to represent a vector in either reference frame, a transformation must be established between the two frames. Various methods exist for performing a transformation between frames, including Euler rotation matrices, quaternion transformations, and angle-axis representation. For the scope of this paper, Euler rotation matrices are used, but their limitations are noted.

Three matrices fully describe the transformation between the body frame and the inertial frame: rotation about the $\hat{b}_z$ axis, rotation about the $\hat{b}_x$ axis, and rotation about the $\hat{b}_y$ axis. Rotation about the $\hat{b}_z$ axis (yaw) is a familiar example, common in 2-D transformations, and can be described by

$$\mathbf{R}_\psi = \begin{bmatrix} \cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \sin(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{1}$$

Similarly, rotation about the $\hat{b}_x$ axis (roll) is described by

$$\mathbf{R}_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sin(\phi) & \cos(\phi) \end{bmatrix}. \tag{2}$$

Finally, rotation about the $\hat{b}_y$ axis (pitch) is described by

$$\mathbf{R}_\theta = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}. \tag{3}$$

While these three matrices fully described the transformation between the two coordinate frames, it is often more convenient to combine them into a single matrix for performing calculations. This final matrix is given by the product of (1)–(3) which yields

$$\mathbf{R}_{\phi,\theta,\psi} = \begin{bmatrix} C_\psi C_\theta & C_\psi S_\phi S_\theta - S_\psi C_\phi & C_\psi C_\phi S_\theta + S_\psi S_\phi \\ S_\psi C_\theta & S_\psi S_\phi S_\theta + C_\psi C_\phi & S_\psi C_\phi S_\theta - C_\psi S_\phi \\ -S_\theta & S_\phi C_\theta & C_\phi C_\theta \end{bmatrix} \tag{4}$$

where $S_x = \sin(x)$ and $C_x = \cos(x)$. A useful property of this rotation matrix is that $\mathbf{R}^{-1}_{\phi,\theta,\psi} = \mathbf{R}^T_{\phi,\theta,\psi}$ which can be used for transforming from the inertial frame to the body frame if needed. However, it should be noted that if $\cos(\theta) = 0$, a singularity occurs in $\mathbf{R}_{\phi,\theta,\psi}$ in which case one degree of freedom is lost. To address this shortcoming, other methods such as quaternion representations are often used when describing aerial robots. However, for the scope of this paper, it is assumed that the quadcopter will not see large angles and therefore will not experience gimbal lock.

C. Newton–Euler Equations

A classic method of deriving the equations of motion in robotics is the use of the Newton–Euler equations. Combined, these equations fully describe both translational and rotational dynamics of a rigid body.

Consider the reference frame B which is attached to the particle at the robot's center of mass as discussed in Section II-A. The velocity of B in E is defined as

$$\vec{v} = \frac{d\vec{p}}{dt} \tag{5}$$

where $\vec{p}$ is the position vector from the origin of E to the origin of B at the robot's center of mass. The translational momentum of B is given by

$$\vec{L} = m\vec{v} \tag{6}$$

where m is the mass of the robot. Newton's second law states that the sum of the external forces acting upon an object equals the time rate of change of its translational momentum. Therefore, the translational dynamics of the robot can be described by

$$\begin{aligned} \sum \vec{F} &= \frac{d}{dt}(\vec{L}) \\ &= \frac{d}{dt}(m\vec{v}). \end{aligned} \tag{7}$$

For the scope of this paper, it is assumed that mass is time-invariant. Taking the derivative in the earth frame yields

$$\begin{aligned} \sum \vec{F}_E &= m\frac{d\vec{v}}{dt_E} \\ &= m\vec{a}_E \end{aligned} \tag{8}$$

where $\vec{a}_E$ is linear acceleration in the earth frame. The derivative can also be taken in B in order to represent the dynamics in the body frame which yields

$$\begin{aligned} \sum \vec{F}_B &= m\left(\frac{d\vec{v}}{dt_B} + \vec{\omega}_B \times \vec{v}_B\right) \\ &= m(\vec{a}_B + \vec{\omega}_B \times \vec{v}_B). \end{aligned} \tag{9}$$

Because B is located at the center of mass of the body, $\vec{\omega}_B$ is zero and (9) simplifies to be

$$\sum \vec{F}_B = m\vec{a}_B. \tag{10}$$

While (10) accurately describes the translational dynamics of the robot, the angular dynamics must still be addressed. The angular momentum of B is defined as

$$\vec{H} = I_{\text{cm}}\vec{\omega} \tag{11}$$

where I_{cm} is the moment of inertia about the robot's center of mass. Euler's second law states that the sum of the torques acting upon an object equals the time rate of change its angular momentum. Therefore, the rotational dynamics are described using

$$\begin{aligned} \sum \vec{\tau} &= \frac{d}{dt}(\vec{H}) \\ &= \frac{d}{dt}(I_{\text{cm}}\vec{\omega}). \end{aligned} \tag{12}$$

Similar to mass, it is assumed that I_{cm} is time-invariant. Therefore, the time derivative of $\vec{H}$, taking into account a rotating frame, is found as

$$\begin{aligned} \sum \vec{\tau}_B &= I_{\text{cm}}\frac{d\vec{\omega}}{dt}_B + \vec{\omega}_B \times I_{\text{cm}}\vec{\omega}_B \\ &= I_{\text{cm}}\vec{\alpha}_B + \vec{\omega}_B \times I_{\text{cm}}\vec{\omega}_B \end{aligned} \tag{13}$$

where $\vec{\alpha}$ is angular acceleration.

It is common to combine (10) and (13) into matrix form for a more compact representation, given by

$$\begin{bmatrix} \sum \vec{F}_B \\ \sum \vec{\tau}_B \end{bmatrix} = \begin{bmatrix} mI_3 & 0 \\ 0 & I_{\text{cm}} \end{bmatrix} \begin{bmatrix} \vec{a}_B \\ \vec{\alpha}_B \end{bmatrix} + \begin{bmatrix} 0 \\ \vec{\omega}_B \times I_{\text{cm}}\vec{\omega}_B \end{bmatrix} \tag{14}$$

where I_3 is a 3×3 identity matrix. This is the classic form of the Newton–Euler equations for a system with B oriented at the system's center of mass, and is the basis for determining the equations of motion specific to the quadcopter platform.

D. Quadcopter Dynamics

The quadcopter platform shown in Fig. 3 is assumed to be symmetric about the $\hat{b}_x$ and $\hat{b}_y$ axis. Thus, the inertia matrix in the body frame is given by

$$I_{\text{cm}} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \tag{15}$$

where I_{xx} and I_{yy} are equal due to symmetry.

Each of the four motors on the quadcopter has a force associated with it, given by

$$\vec{T}_i = -k_T\Omega_i^2\hat{b}_z \tag{16}$$

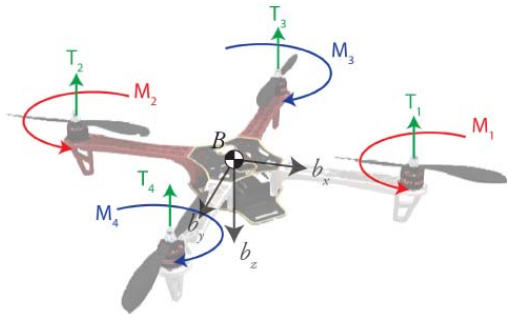

Fig. 3. Quadcopter is symmetric about both $\hat{b}_x$ and $\hat{b}_y$ axis. Each motor produces a force, $\vec{T}_i$ and moment, $\vec{M}_i$, which are functions of the motor's angular velocity, Ω_i.

where $\vec{T}_i$ is the force (or thrust) provided by the ith motor, Ω_i is the angular velocity of the motor, and k_T is a constant which is a function of the specific motor and propeller used. For the scope of this paper, it is assumed that the density of air remains constant and that roll and pitch are small angles. Using this, sum of the forces is found to be

$$\sum \vec{F} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & mg - k_T \sum_{i=1}^{4} \Omega_i^2 \end{bmatrix} \begin{bmatrix} \hat{b}_x \\ \hat{b}_y \\ \hat{b}_z \end{bmatrix} \tag{17}$$

$$\sum \vec{\tau} = \begin{bmatrix} k_T(\Omega_3^2 - \Omega_4^2)l & 0 & 0 \\ 0 & k_T(\Omega_1^2 - \Omega_2^2)l & 0 \\ 0 & 0 & -k_M(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) \end{bmatrix} \begin{bmatrix} \hat{b}_x \\ \hat{b}_y \\ \hat{b}_z \end{bmatrix} \tag{18}$$

$$\begin{aligned} \begin{bmatrix} \vec{a}_x \\ \vec{a}_y \\ \vec{a}_z \\ \vec{\alpha}_x \\ \vec{\alpha}_y \\ \vec{\alpha}_z \end{bmatrix} &= \begin{bmatrix} \frac{1}{m} & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{m} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{I_{xx}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{I_{yy}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \\ &\quad \times \left(\begin{bmatrix} 0 \\ 0 \\ mg - k_T \sum_{i=1}^{4} \Omega_i^2 \\ k_T(\Omega_3^2 - \Omega_4^2)l \\ k_T(\Omega_1^2 - \Omega_2^2)l \\ -k_M(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ \dot{\theta}\dot{\psi}(I_{zz} - I_{yy}) \\ \dot{\phi}\dot{\psi}(I_{xx} - I_{zz}) \\ \dot{\theta}\dot{\phi}(I_{yy} - I_{xx}) \end{bmatrix} \right) \\ &= \begin{bmatrix} 0 \\ 0 \\ g - \frac{k_T}{m} \sum_{i=1}^{4} \Omega_i^2 \\ \frac{1}{I_{xx}}(k_T(\Omega_3^2 - \Omega_4^2)l - \dot{\theta}\dot{\psi}(I_{zz} - I_{yy})) \\ \frac{1}{I_{yy}}(k_T(\Omega_1^2 - \Omega_2^2)l - \dot{\phi}\dot{\psi}(I_{xx} - I_{zz})) \\ \frac{1}{I_{zz}}(-k_M(\Omega_1^2 + \Omega_2^2 - \Omega_3^2 - \Omega_4^2) - \dot{\theta}\dot{\phi}(I_{yy} - I_{xx})) \end{bmatrix}. \end{aligned} \tag{19}$$

Similar to thrust, each motor also has an associated moment, given by

$$\vec{M}_i = -k_M \Omega_i^2 \hat{b}_z \tag{20}$$

where $\vec{M}_i$ is the moment generated by the ith motor, and k_M is a constant which is again a function of the specific motor and propeller used. In addition to the moments generated by the motors themselves, there are torque contributions from the moment arms produced by the motor's forces. The sum of the torques can be found in (18), where l is the length of the quadcopter's arms.

From (17) and (18), it is apparent that the linear translational dynamics are closely coupled with the rotational dynamics. This is expected because a quadcopter is an underactuated system, having only four actuators and six degrees of freedom.

Using the result of (17) and (18) in the Newton–Euler equations, the dynamics of the system in the body frame, B, are described by (19), assuming small roll and pitch angles. These equations fully describe the motion of the quadcopter platform in the body coordinates, and can be used for simulating the dynamics of the platform to evaluate controller performance.

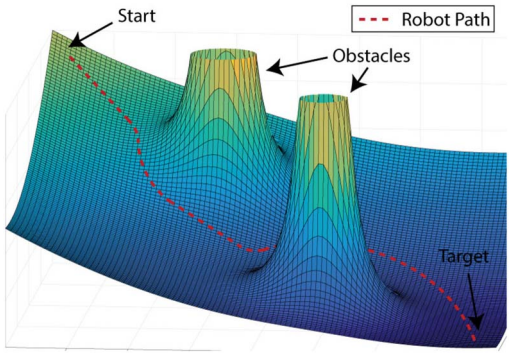

Fig. 4. Example of a traditional potential field which can be used for navigating toward a target while avoiding multiple obstacles.

III. Controller Design and Stability

Because of their simplicity and elegance, PFCs are often used for navigation of ground robots [37]–[39]. Potential fields are aptly named, because they use attractive and repulsive potential equations to draw the drone toward a goal (attractive potential) or push it away from an obstacle (repulsive potential). For example, imagine a stretched spring which connects a drone and a target. Naturally, the spring draws the drone to the target location.

Conveniently, potential fields for both attractive and repulsive forces can be summed together, to produce a field such as the one shown in Fig. 4. This figure illustrates how a robot can navigate toward a target location while simultaneously avoiding obstacles in its path.

A. Controller Design

Denote $\vec{p}_d = [x_d, y_d, z_d]^T$ and $\vec{p}_t = [x_t, y_t, z_t]^T$ as the position vector of drone and target, respectively. The relative distance vector between the drone and the target is then

$$\begin{aligned} \vec{p}_{dt} &= \left[x_{dt}, y_{dt}, z_{dt}\right]^T \\ &= \left[x_d, y_d, z_d\right]^T - \left[x_t, y_t, z_t\right]^T. \end{aligned} \tag{21}$$

Traditionally, potential forces work in the x, y, and z spatial dimensions, and are defined by a quadratic function given by

$$U_{\text{att}1}(\vec{p}_d, \vec{p}_t) = \frac{1}{2}\lambda_1 \|\vec{p}_{dt}\|^2 \tag{22}$$

where λ_1 is positive scale factor, and $\|\vec{p}_{dt}\|$ is the magnitude of the relative distance between the drone and the target, which is given by

$$\|\vec{p}_{dt}\| = \sqrt{(x_{dt})^2 + (y_{dt})^2 + (z_{dt})^2}. \tag{23}$$

As shown in Fig. 4, the target location is always a minimum, or basin, of the overall potential field. Therefore, in order to achieve the target location, the UAS should always move "downhill." The direction and magnitude of the desired movement can be computed by finding the negative gradient of the potential field, given by

$$\begin{aligned}\vec{v}_d^{\,\text{att}^1}(\vec{p}_d, \vec{p}_t) &= -\nabla U_{\text{att}^1}(\vec{p}_d, \vec{p}_t) \\ &= -\frac{\partial U_{\text{att}^1}}{\partial x}\hat{i} - \frac{\partial U_{\text{att}^1}}{\partial y}\hat{j} - \frac{\partial U_{\text{att}^1}}{\partial z}\hat{k} \\ &= -\nabla\left(\frac{1}{2}\lambda_1 \|\vec{p}_{dt}\|^2\right) \\ &= -\frac{1}{2}\lambda_1 \nabla\left(\vec{x}_{dt}^2 + \vec{y}_{dt}^2 + \vec{z}_{dt}^2\right) \\ &= -\lambda_1(\vec{x}_{dt} + \vec{y}_{dt} + \vec{z}_{dt}) \\ &= -\lambda_1(\vec{p}_d - \vec{p}_t)\end{aligned} \tag{24}$$

where $\vec{v}_d^{\,\text{att}^1}$ is the desired velocity due to the attractive position potential.

This is the classic form of a simple attractive PFC. However, this does not yet take into account obstacles or other sources of repulsive potential. The repulsive potential is proportional to the inverse square of the distance between the drone and the obstacle and is given by

$$U_{\text{rep}^1}(\vec{p}_d, \vec{p}_o) = \frac{1}{2}\eta_1 \frac{1}{\|\vec{p}_{\text{do}}\|^2} \tag{25}$$

where η_1 is positive scale factor, and $\|\vec{p}_{\text{do}}\|$ is the magnitude of the relative distance between the drone and the obstacle.

To find the desired velocity, again take the gradient of the potential field which yields

$$\begin{aligned}\vec{v}_d^{\,\text{rep}^1}(\vec{p}_d, \vec{p}_o) &= -\nabla U_{\text{rep}^1}(\vec{p}_d, \vec{p}_o) \\ &= -\frac{\partial U_{\text{rep}^1}}{\partial x}\hat{i} - \frac{\partial U_{\text{rep}^1}}{\partial y}\hat{j} - \frac{\partial U_{\text{rep}^1}}{\partial z}\hat{k} \\ &= -\nabla\left(\frac{1}{2}\eta_1 \frac{1}{\|\vec{p}_{\text{do}}\|^2}\right) \\ &= -\frac{1}{2}\eta_1 \nabla\left(\frac{1}{\vec{x}_{\text{do}}^2 + \vec{y}_{\text{do}}^2 + \vec{z}_{\text{do}}^2}\right) \\ &= \eta_1 \frac{\vec{x}_{\text{do}} + \vec{y}_{\text{do}} + \vec{z}_{\text{do}}}{\left(\vec{x}_{\text{do}}^2 + \vec{y}_{\text{do}}^2 + \vec{z}_{\text{do}}^2\right)^2} \\ &= \eta_1 \frac{\vec{p}_{\text{do}}}{\|\vec{p}_{\text{do}}\|^4}\end{aligned} \tag{26}$$

where $\vec{v}_d^{\,\text{rep}^1}$ is the desired velocity due to the repulsive position potential.

It is important to note that the repulsive potential is designed to have no effect when the drone is more than a set distance away from the obstacle, P^*, as shown in Fig. 5.

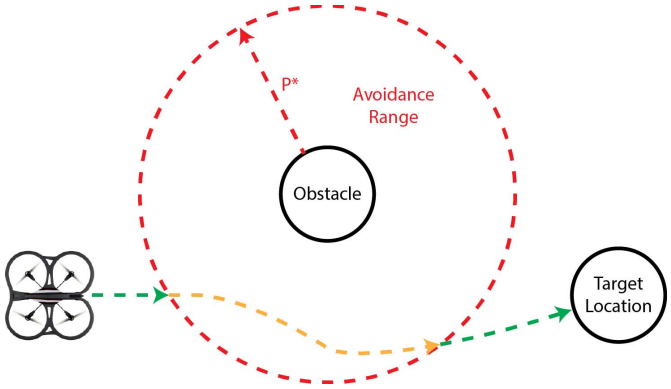

Fig. 5. Repulsive force due to an obstacle should be zero when the drone is sufficiently far from it. In this case, the limit is P^*. While outside of this limit, the drone is unaffected by the obstacle's repulsive potential. However, as soon as the relative distance becomes less than P^*, the drone takes avoidance action.

Therefore, the velocity due to repulsive potentials becomes

$$\vec{v}_d^{\,\text{rep}^1}(\vec{p}_d, \vec{p}_o) = \begin{cases} \eta_1 \frac{\vec{p}_{\text{do}}}{\|\vec{p}_{\text{do}}\|^4}, & \|\vec{p}_{\text{do}}\| \le P^* \\ 0, & \|\vec{p}_{\text{do}}\| > P^*. \end{cases} \tag{27}$$

A complete traditional PFC is the sum of (24) and (27) which yields (28), where n is the number of obstacles present in the environment

$$\begin{aligned}&\vec{v}_d^{\,\text{PFC}}(\vec{p}_d, \vec{p}_t, \vec{p}_o) \\ &= \begin{cases} -\lambda_1(\vec{p}_t - \vec{p}_d) + \sum_{i=0}^{n} \eta_1 \frac{\vec{p}_{\text{do}}^{\,i}}{\|\vec{p}_{\text{do}}^{\,i}\|^4}, & \|\vec{p}_{\text{do}}^{\,i}\| \le P^* \\ -\lambda_1(\vec{p}_t - \vec{p}_d), & \|\vec{p}_{\text{do}}^{\,i}\| > P^*. \end{cases}\end{aligned} \tag{28}$$

This controller enables a ground robot to track stationary or dynamic targets, while avoiding any obstacles in its path. However, when applied to an agile, aerial system such as a quadcopter, the controller's performance is quite poor as shown later in simulations presented in Section IV.

B. Extended Potential Field Controller

Because the potential field methods presented above are developed for ground robots, they do not address many of the factors that must be accounted for when designing a controller for aerial systems. For example, drones move very quickly and are inherently unstable which means they cannot simply move to a particular location and stop moving. They are consistently making fine adjustments to their position and velocity.

In order to account for factors unique to aerial platforms, this paper presents an extended PFC (ePFC) which utilizes the same concepts found in a traditional PFC, but applied to relative velocities rather than positions. Now, considering that the system is tracking a dynamic target, the desired velocity is that of the target. In this case, the attractive potential is defined as the quadratic function given by

$$U_{\text{att}^2}(\vec{v}_d, \vec{v}_t) = \frac{1}{2}\lambda_2 \|\vec{v}_{dt}\|^2 \tag{29}$$

where λ_2 is positive scale factor, and $\|v_{dt}\|$ is the magnitude of the relative velocity between the drone velocity, v_d, and the

target velocity, v_t, which is given by

$$\|\vec{v}_{dt}\| = \sqrt{(\dot{x}_{dt})^2 + (\dot{y}_{dt})^2 + (\dot{z}_{dt})^2}. \tag{30}$$

As in the traditional PFC the relative velocity potential should be minimized, thus resulting in a matched velocity between the drone and the target. Similar to the traditional controller, the desired velocity of the drone is found by calculating the negative gradient, which is

$$\begin{aligned}\vec{v}_d^{\,\text{att}^2}(\vec{v}_d, \vec{v}_t) &= -\nabla U_{\text{att}^2}(\vec{v}_d, \vec{v}_t) \\ &= -\frac{\partial U_{\text{att}^2}}{\partial \dot{x}}\hat{i} - \frac{\partial U_{\text{att}^2}}{\partial \dot{y}}\hat{j} - \frac{\partial U_{\text{att}^2}}{\partial \dot{z}}\hat{k} \\ &= -\nabla\left(\frac{1}{2}\lambda_2\|\vec{v}_{dt}\|^2\right)\right) \\ &= -\frac{1}{2}\lambda_2\nabla\left(\dot{\bar{x}}_{dt}^2 + \dot{\bar{y}}_{dt}^2 + \dot{\bar{z}}_{dt}^2\right) \\ &= -\lambda_2\left(\dot{\bar{x}}_{dt} + \dot{\bar{y}}_{dt} + \dot{\bar{z}}_{dt}\right) \\ &= -\lambda_2(\vec{v}_{dt}).\end{aligned} \tag{31}$$

It is desirable that the drone and an obstacle should not maintain the same velocity, therefore the repulsive velocity potential between the drone and an obstacle is designed to be an inverse quadratic as in (25), given by

$$U_{\text{rep}^2}(\vec{v}_d, \vec{v}_o) = \frac{1}{2}\eta_2\frac{1}{\|\vec{v}_{\text{do}}\|^2} \tag{32}$$

where η_2 is positive scale factor, and $\|\vec{v}_{\text{do}}\|$ is the magnitude of the relative velocity between the drone velocity, $\vec{v}_d$, and the obstacle velocity, $\vec{v}_o$. The corresponding velocity for this potential function is found by

$$\begin{aligned}\vec{v}_d^{\,\text{rep}^2}(\vec{v}_d, \vec{v}_o) &= \nabla U_{\text{rep}^2}(\vec{v}_d, \vec{v}_o) \\ &= \frac{\partial U_{\text{rep}^2}}{\partial \dot{x}}\hat{i} + \frac{\partial U_{\text{rep}^2}}{\partial \dot{y}}\hat{j} + \frac{\partial U_{\text{rep}^2}}{\partial \dot{z}}\hat{k} \\ &= -\nabla\left(\frac{1}{2}\eta_2\frac{1}{\|\vec{v}_{\text{do}}\|^2}\right) \\ &= -\frac{1}{2}\eta_2\nabla\left(\frac{1}{\dot{\bar{x}}_{\text{do}}^2 + \dot{\bar{y}}_{\text{do}}^2 + \dot{\bar{z}}_{\text{do}}^2}\right) \\ &= \eta_2\frac{\vec{v}_d - \vec{v}_o}{\left(\dot{\bar{x}}_{\text{do}}^2 + \dot{\bar{y}}_{\text{do}}^2 + \dot{\bar{z}}_{\text{do}}^2\right)^2} \\ &= \eta_2\frac{\vec{v}_{\text{do}}}{\|\vec{v}_{\text{do}}\|^4}.\end{aligned} \tag{33}$$

It should be noted that if the obstacle is stationary, then its velocity is zero. In this special case, the repulsive field in (32) is designed to have no effect on the drone's motion. Therefore, the velocity found in (33) becomes

$$\vec{v}_d^{\,\text{rep}^2}(\vec{v}_d, \vec{v}_o) = \begin{cases}\eta_2\frac{\vec{v}_d - \vec{v}_o}{\|\vec{v}_{\text{do}}\|^4}, & \|\vec{v}_o\| \neq 0 \\ 0, & \|\vec{v}_o\| = 0.\end{cases} \tag{34}$$

Finally, the relative distance between the drone and obstacle, $\vec{p}_{\text{do}}$, is revisited. In the traditional PFC presented previously, $\vec{p}_{\text{do}}$ was used as the basis for a repulsive potential. However, no thought is given to the time rate of change of the magnitude of $\vec{p}_{\text{do}}$. If a situation in which the drone is moving away from the obstacle is considered, then $\|\dot{p}_{\text{do}}\| \geq 0$ in which case no avoidance action needs to be taken, even if the drone is within the avoidance range. As illustrated in Fig. 6, the controller is able to ignore the effect of the obstacle sooner, and therefore can take a more direct route, thus saving time.

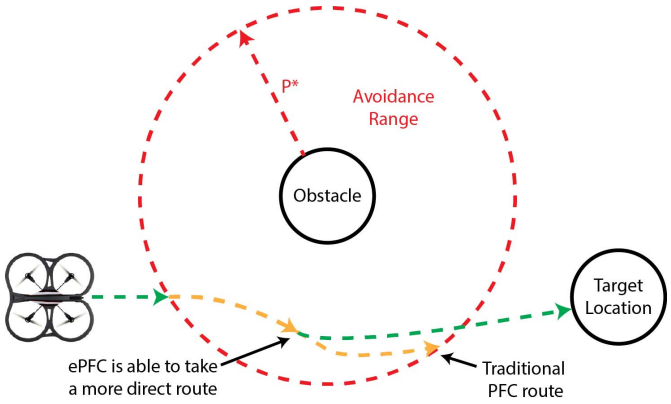

Fig. 6. By taking into account the time rate of change of $\|\vec{p}_{\text{do}}\|$, the controller is able to ignore repulsive effects from the obstacle if the drone is moving away from the obstacle ($\|\dot{p}_{\text{do}}\| \geq 0$). This results in a more direct route to the target, thus saving time.

Furthermore, the drone should take evasive action if $\|\dot{p}_{\text{do}}\| < 0$, when the distance between the two is decreasing. Therefore, a final control effort is designed as

$$\vec{v}_d^{\,\text{rep}^3}(\vec{p}_d, \vec{p}_o) = \begin{cases}-\eta_3\|\dot{p}_{\text{do}}\|\frac{\vec{p}_{\text{do}}}{\|\vec{p}_{\text{do}}\|}, & \|\dot{p}_{\text{do}}\| < 0 \\ 0, & \|\dot{p}_{\text{do}}\| \geq 0\end{cases} \tag{35}$$

where η_3 is positive scale factor.

Summing the velocities in (31), (34), and (35) with the traditional controller (28) yields the full form of the ePFC, which is

$$\begin{aligned}\vec{v}_d^{\,\text{ePFC}} = \vec{v}_d^{\,\text{PFC}} - \lambda_2(\vec{v}_d - \vec{v}_t) + \sum_{i=0}^{n}\eta_2\frac{\vec{v}_d - \vec{v}_o^{\,i}}{\|\vec{v}_{\text{do}}^{\,i}\|^4} \\ - \sum_{i=0}^{n}\eta_3\|\dot{p}_{\text{do}}^{\,i}\|\frac{\vec{p}_{\text{do}}^{\,i}}{\|\vec{p}_{\text{do}}^{\,i}\|}\end{aligned} \tag{36}$$

where n is the number of obstacles present, $\|v_o^i\| \neq 0$, $\|\dot{p}_{\text{do}}^{\,i}\| < 0$, and the same conditions discussed previously apply to v_d^{PFC}.

Finally, the velocity found in (36) must be transformed into the body coordinate system of the drone and is found to be

$$\vec{v}_{d,\text{body}}^{\,\text{ePFC}} = \vec{v}_d^{\,\text{ePFC}} * \begin{bmatrix}\cos(\psi) & -\sin(\psi) & 0 \\ \sin(\psi) & \sin(\psi) & 0 \\ 0 & 0 & 1\end{bmatrix} \tag{37}$$

where ψ is the yaw angle of the drone around the body z-axis.

This controller seeks out a moving target, and also avoids obstacles that are in close proximity.

C. Stability Analysis

To analyze the convergence of the proposed velocity controller (36) for the drone, the Lyapunov theory is used. Consider a positive definite Lyapunov function as follows:

$$L = U_{\text{att}} = \frac{1}{2}\lambda_1\|\vec{p}_{dt}\|^2 + \frac{1}{2}\lambda_2\|\vec{v}_{dt}\|^2. \tag{38}$$

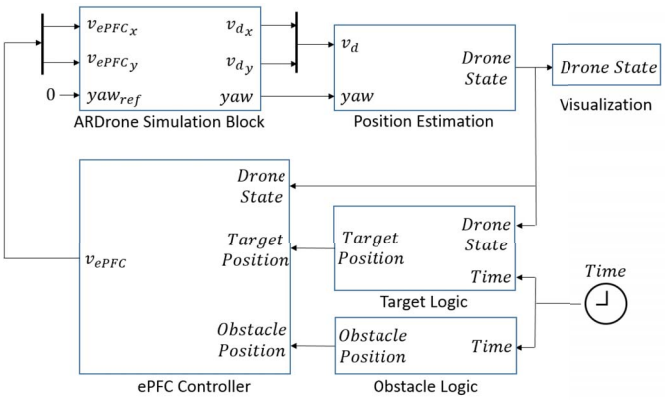

Fig. 7. Simulink model used includes a state space representation from the ARDrone Simulink Development Kit, as well as custom blocks for the ePFC controller described in this paper.

This function represents the artificial potentials of the controller. Since (38) is positive definite, its Lie derivative is given by

$$\begin{aligned} L^* &= \frac{\partial L}{\partial \vec{p}_{dt}}\vec{v}_{dt} + \frac{\partial L}{\partial \vec{v}_{dt}}\vec{a}_{dt} \\ &= \lambda_1 \|\vec{p}_{dt}\|\vec{v}_{dt} + \lambda_2 \|\vec{v}_{dt}\|\vec{a}_{dt} \end{aligned} \tag{39}$$

where $\vec{a}_{dt}$ is the relative acceleration between the drone acceleration and the target acceleration.

Note that the relative velocity between the drone and the target is designed following the direction of the negative gradient of $U_{\text{att}}(p_{dt})$ with respect to p_{dt} as in (24). From (31), obtain

$$\begin{aligned} \vec{a}_{dt} = \dot{\vec{v}}_{dt} &= \frac{d}{dt}\left(-\frac{1}{2}\lambda_2 \nabla\left(\dot{\vec{x}}_{dt}^2 + \dot{\vec{y}}_{dt}^2 + \dot{\vec{z}}_{dt}^2\right)\right) \\ &= \frac{d}{dt}(-\lambda_2 \|\vec{v}_{dt}\|) \\ &= -\lambda_2 \left\| \frac{\vec{v}_{dt}(t) - \vec{v}_{dt}(t-1)}{\Delta_t} \right\| \end{aligned} \tag{40}$$

where Δ_t is a time step. Hence, substituting v_{dt} given by (24) and a_{dt} given by (40) into (39)

$$L^* = -\left[\lambda_1^2 \|\vec{p}_{dt}\|^2 + \lambda_2^2 \|\vec{v}_{dt}\| \left\| \frac{\vec{v}_{dt}(t) - \vec{v}_{dt}(t-1)}{\Delta_t} \right\|\right]. \tag{41}$$

It can be easily seen that $L^* < 0$ since $\|\vec{p}_{dt}\|$, $\|\vec{v}_{dt}\|$, and $\|[(\vec{v}_{dt}(t) - \vec{v}_{dt}(t-1))/\Delta_t]\|$ are positive. This means that the proposed controller is stable, and the drone is able to track a moving target.

IV. Simulation

A. MATLAB Environment

In order to validate the developed controller, the system was simulated using a MATLAB Simulink model. The state space representation of the ARDrone's platform dynamics are take from the ARDrone Simulink Development Kit [40]. The complete Simulink model shown in Fig. 7 demonstrates how the ePFC controller uses feedback information from the ARDrone simulation and position estimator blocks. The output of the ARDrone simulation block is simply the velocity of the drone, and the position estimator uses an integrator with zero initial conditions to calculate position.

TABLE I
Simulation Waypoints

Waypoint	X Coordinate [m]	Y Coordinate [m]
1	2.5	-1
2	2.5	1
3	-2.5	1
4	-2.5	-1

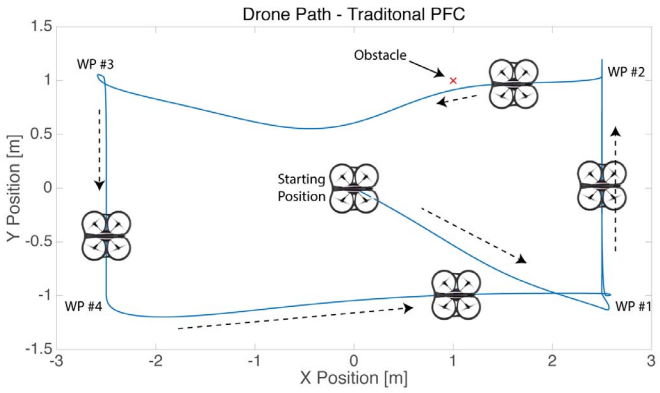

Fig. 8. Traditional PFC is simulated on the ARDrone, with poor results. Because drones cannot stop instantaneously like ground robots, the drone often overshoots the desired waypoint. For reference, the drone takes approximately 35 s to complete a full loop of the course.

The desired path that the drone is to take is outlined in Table I. A virtual obstacle is placed at $(1, 1)$ which places it immediately in the path of the drone between waypoints 2 and 3. The drone is allowed 2 s at each waypoint in an attempt to let it settle before moving on to the next waypoint.

B. Simulation Results

First, a traditional PFC was simulated, and the resulting path is shown in Fig. 8. The performance of the traditional PFC was poor as expected, because aerial drones have very different dynamics than their ground counterparts. Using the traditional PFC, the drone overshoots the desired waypoint, and while it does avoid the obstacle at $(1, 1)$ it is not by much. The drone completed a full loop in approximately 35 s.

Next, the ePFC is tested using the same path and obstacle position. The results shown in Fig. 9 demonstrate the effectiveness of the new controller. The drone does not overshoot the desired waypoints and avoids the obstacle by a larger margin, while completing the course in a shorter amount of time than the traditional controller.

As outlined in Table II, the ePFC controller has zero overshoot, and has a settling time of approximately 5 s. This is a large improvement over the traditional controller which overshoots by up to 19% and takes nearly 6 s to settle. It is clear that the proposed controller is more appropriate for use on an aerial drone than the traditional PFC. It is also important to note that at the furthest point (waypoint 4) the drone is approximately 4 m from the obstacle which is still within the avoidance range P^* for the simulation. The ePFC outperforms

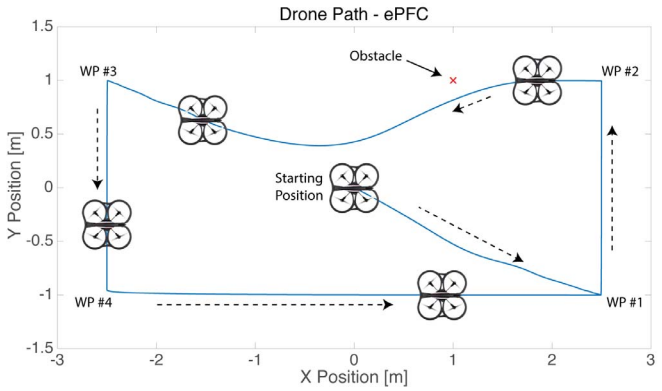

Fig. 9. Using the ePFC, the drone is able to complete the course without overshooting the target waypoints, and avoids the obstacle by a larger margin than the traditional controller. Because the drone does not overshoot the target, it is able to complete the course in a shorter amount of time compared to the traditional controller.

TABLE II
SIMULATION CONTROLLER EVALUATION

Controller	Overshoot [%]	Settling Time [sec]
Traditional PFC	>19%	6
ePFC	0%	5

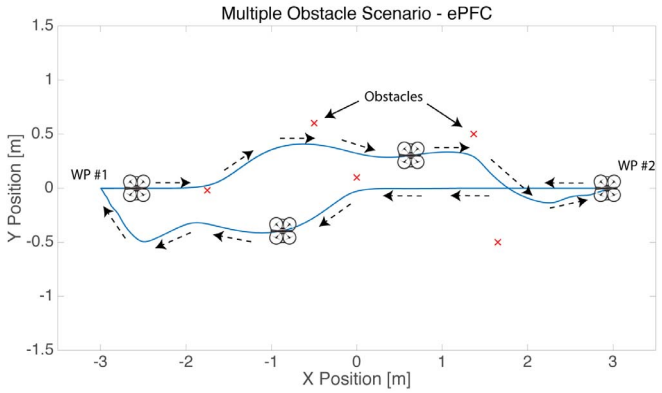

Fig. 10. More complex simulation was performed with multiple obstacles placed throughout the environment. The drone is able to successfully navigate between waypoints without colliding with a single obstacle, thus demonstrating its effectiveness in multiobstacle scenarios.

the traditional PFC because it takes into account the changing relative distance between the drone and the obstacle behind it. Since the relative distance between the drone and the obstacle is increasing, the repulsive obstacle potential has no effect as mentioned in (35). The traditional PFC does not take this into account and therefore is being pushed past the target even though it is already moving away from it.

In addition to the comparison between the tradition PFC and the ePFC, a more complex simulation was performed which included several obstacles placed at random throughout the environment. The results shown in Fig. 10 demonstrate the that the ePFC is very effective in multiobstacle scenarios, and it successfully navigates between waypoints without colliding with a single obstacle.

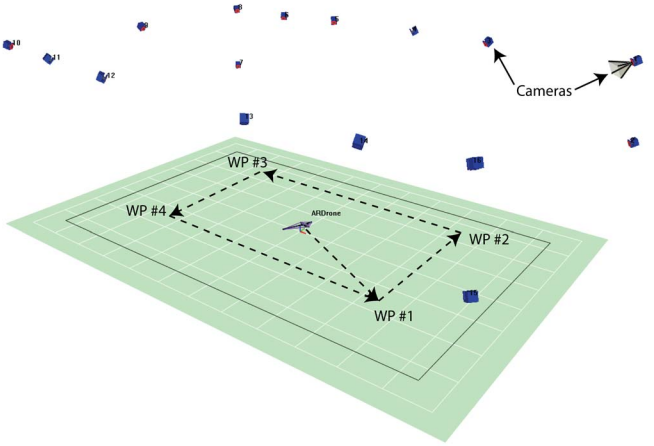

Fig. 11. Motion analysis cortex software gives the user a real-time, visual 3-D representation of the environment including camera locations and any objects sensed by the system [41].

V. EXPERIMENTAL RESULTS

This section presents the experimental setup used to implement the proposed controller. In addition, the results from implementation are presented and the performance of the proposed controller is discussed.

A. Experimental Setup

The experimental platform chosen to implement the ePFC is the ARDrone 2.0 quadcopter shown in Fig. 1. This platform was chosen for its ease of communication—over a WiFi connection—as well as the safety provided by the foam hull. Additionally, the ARDrone requires little to no setup and spare parts are readily available in case of crashes. The ARDrone 2.0 can be equipped with a 1500-mAh battery which yields flight times up to 18 min. Large batteries and long flight times are very advantageous in a testing environment because it allows for more uninterrupted tests and less downtime recharging batteries. The ARDrone 2.0 is also equipped with a 1-GHz 32-bit ARM Cortex A8 processor, 1-GB DDR2 RAM, and runs Linux. This means that the developed controller can be implemented on-board the drone in future work.

Sixteen Motion Analysis Kestrel cameras located throughout the testing space provide the position and orientation of the drone, target (if not virtual), and any obstacles present. The Cortex software suite provides a visual representation of the environment as shown in Fig. 11 as well as sending data over a network connection for use by external programs.

In order to control the drone and display its location along with the target and any obstacles present, the MATLAB GUI shown in Fig. 12 was created. It allows the user to determine when the drone takes off, lands, or tracks the target. This GUI is critical in efficient testing of the drone. Additionally, for the safety of the drone, if the controller does not behave as expected the user can request that the drone simply hover in place to avoid fly-aways.

The overview of the experimental setup shown in Fig. 13 demonstrates the feedback loop implemented. The Motion Analysis external tracking system is used for localization of

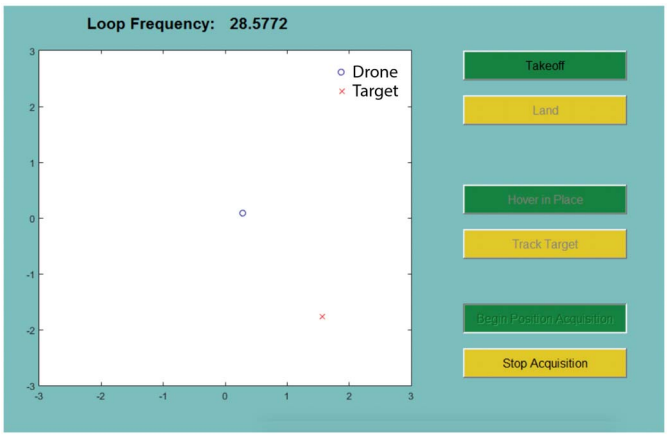

Fig. 12. MATLAB GUI was created to show the positions of the drone, target, and any obstacles present. It also allows the user to control when the drone takes off, lands, tracks the target, or simply hovers in place.

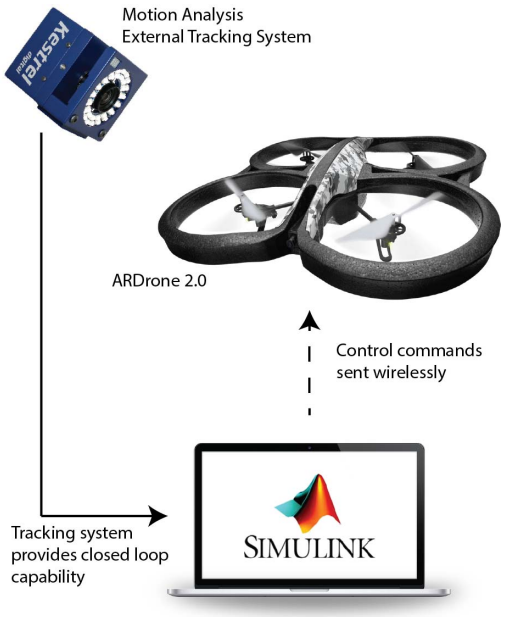

Fig. 13. Experimental setup for this paper includes a motion analysis external tracking system with 16 cameras which provides the position of the drone and obstacles in the environment. The same Simulink model used in Section IV provides control commands to the ARDrone over a wireless connection.

the drone and obstacles in real time. The position information is used by the same Simulink model shown in Section IV which controls the ARDrone over a wireless connection.

B. Experimental Results and Discussion

Having validated the controller using the Simulink simulation, it was then implemented on the actual ARDrone. Several experiments were formed, in the order outlined in Table III.

Because the simulation showed a clear improvement in performance between the tradition PFC and the developed ePFC, the traditional controller was not tested on the experimental platform. Instead, the ePFC was immediately implemented in the experiments.

TABLE III
EXPERIMENTAL TESTS

Test Number	Target	Obstacle(s)
1	1 - Static	0 - N/A
2	1 - Static	1 - Static
3	1 - Dynamic Square	0 - N/A
4	4 - Static Waypoints	1 - Static
5	1 - Dynamic No Pattern	0 - N/A

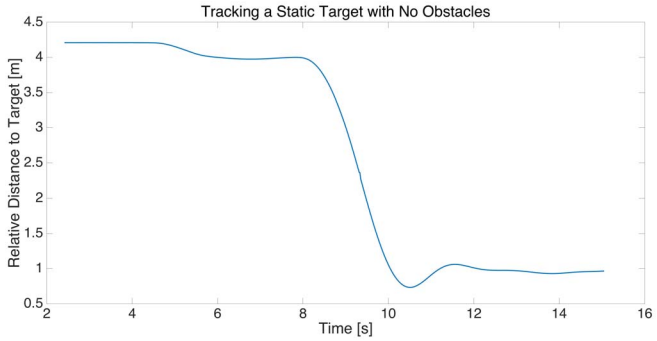

Fig. 14. Results of tracking a static target with no obstacles are very good. With an initial condition of approximately 3.2 m, the drone achieves position in under 5 s.

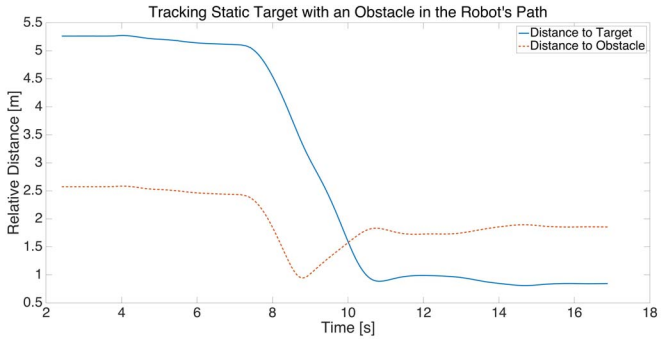

Fig. 15. Results of tracking a static target with an obstacle in the way demonstrates the controllers effectiveness at avoiding collisions. As expected, the drone's position relative to the obstacle decreases, but the drone takes avoidance action and never gets closer than one meter away from the obstacle.

In the first test, the drone was placed approximately 4.2 m from the target's location. Because the target wand is often held by a human, the drone was requested to fly to 1 m away from the target location to avoid collision with someone holding the wand. The drone's response shown in Fig. 14 demonstrates the capability of the drone to achieve a goal position effectively. Starting at approximately 7.75 s, the drone enters an autonomous mode, and achieves stable hover 1 m away from the target in approximately 5 s. It is important to note that while the drone did overshoot its goal location, it did not overshoot enough to get close to hitting the target. The closest that the drone got to the target was just under 0.75 m. In the second experiment, the drone was placed approximately 5.2 m away from the target wand, and an obstacle was located in the path lying directly between the drone and the target. Similar to the first test, the drone's mission was to fly to within 1 m of the target, this time while avoiding the obstacle and

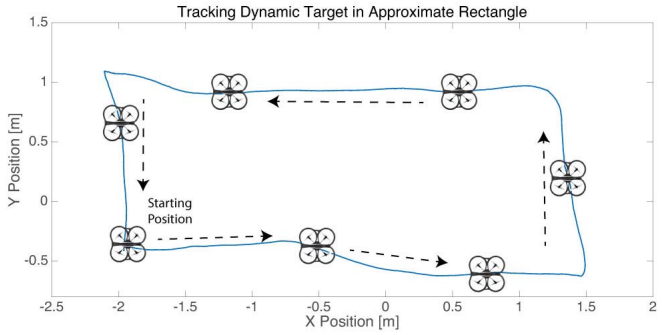

Fig. 16. In the third experiment, the drone tracks a target which moves in an approximate rectangle. As shown, the drone does track, but because the desired trajectory is human-controlled the reference is not perfect. Test number four addresses this imperfection by using set virtual waypoints.

TABLE IV
EXPERIMENTAL WAYPOINTS

Waypoint	X Coordinate [m]	Y Coordinate [m]
1	1.5	-0.5
2	1.5	0.5
3	-1.5	0.5
4	-1.5	-0.5

still achieving the task. As the drone begins moving toward the target, it also moves toward the obstacle. Because of the repulsive forces generated by the relative position and velocity with respect to the obstacle, the drone is elegantly pushed around the obstacle and still makes it to the target location. Fig. 15 shows the results of this test, demonstrating that the drone maintains a safe distance from the obstacle (1 m minimum) and also achieves the goal.

In the third test, the drone was instructed to follow the target wand as it moved in an approximate rectangle around the laboratory. The results shown in Fig. 16 illustrate the path of the drone as it follows the target through the pattern. As shown, the drone does in fact track the rectangle as instructed. Because the path of the target was moved manually by a person holding the wand, the target trajectory is not a perfect rectangle. Therefore, the next experiment establishes a perfect rectangle using virtual waypoints.

In test number four, virtual waypoints like those used in simulation are used to demonstrate the ability of the drone to navigate a course and avoid obstacles. The waypoints used for this test are outlined in Table IV.

The results from the fourth experiment shown in Figs. 17 and 18 demonstrate that the drone successfully reaches each waypoint, and also avoids the obstacle in its path between waypoints 2 and 3.

To quantify the controller performance, the error in response to a waypoint change, or step input, is shown in Fig. 19. The x-axis error is chosen as the worst case scenario in the experiment, having a step input of over 2.5 m versus only 1 m on the y-axis.

The controller's performance is quite good to step inputs, with an approximate settling time of 5.5 s, and a percent overshoot of only 1.8%. A comparison between the simulated and experimental results is outlined in Table V. While the experimental results do have a slightly longer settling time, and more overshoot, this is not surprising. In a real-world application, the controller is subject to disturbances such as ground effects from propeller wash, since the drone is operating close to the ground and desks.

As the drone approaches the obstacle, the repulsive potential pushes the drone around it as expected. In this experiment, the drone avoids the obstacle by a margin of approximately 0.5 m.

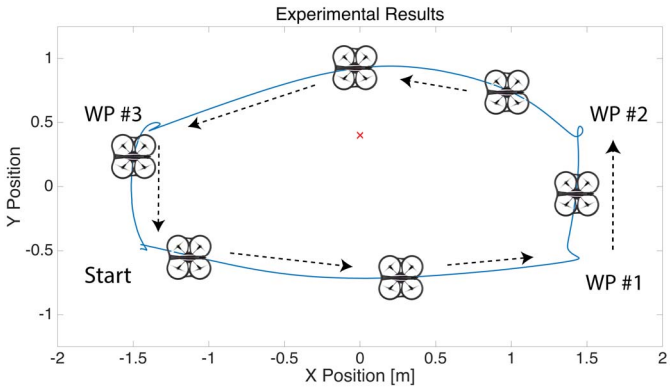

Fig. 17. To test the ePFC experimentally, the drone follows waypoints similar to those in the simulation. The drone successfully reaches each waypoint and avoids the obstacle in its path between waypoints 2 and 3.

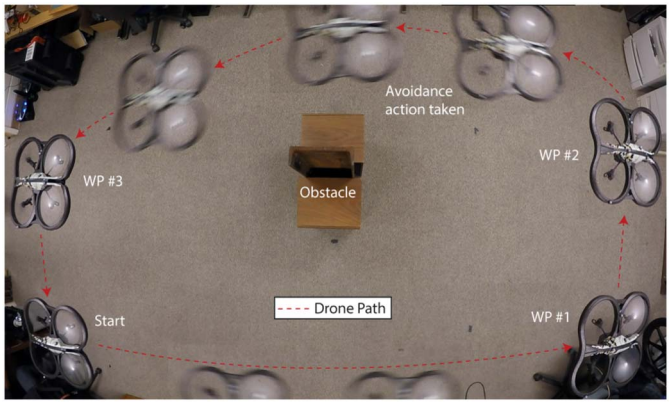

Fig. 18. Stills from a video demonstrate the drone taking avoidance action while tracking moving waypoints.

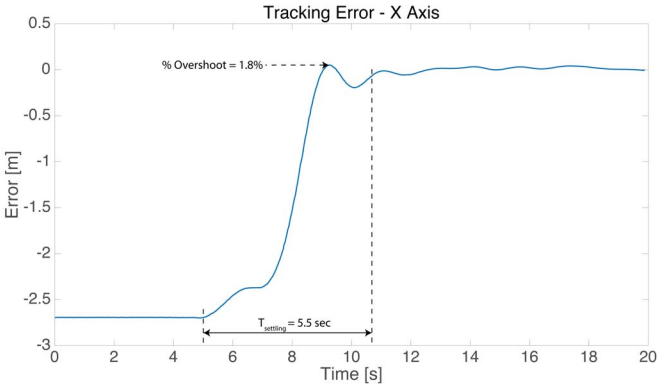

Fig. 19. Error in response to a waypoint change, or a step input, results in a settling time of approximately 5.5 s and a percent overshoot of only 1.8%. The x-axis was chosen because the step input for this direction was the largest, at over 2.5 m, whereas the y input is only 1 m.

TABLE V
SIMULATION VERSUS EXPERIMENTAL EVALUATION

Experiment	Overshoot [%]	Settling Time [sec]
Simulated ePFC	0%	5
Experimental ePFC	1.8%	5.5

Fig. 20. Third experiment demonstrates the drone tracking a dynamic target, while maintaining the proper heading to always face the target.

Thus, this demonstrates that the drone can successfully avoid obstacles.

In addition to tracking static targets and virtual waypoints, a final test is performed in which the ARDrone is commanded to follow the target as it moves about the laboratory environment in an arbitrary pattern. During this experiment, the drone must maintain a safe distance at all times and should always face the target. This task was performed several times to evaluate the performance. In each of the tests the drone successfully completes the task. Even under extreme circumstances (e.g., very fast maneuvers) the drone is able to recover and maintain the desired behavior. Fig. 20 shows frames from a video [42] taken of the drone performing this task. In the video it can clearly be seen that the drone follows the target around while always maintaining the proper heading to face the target. For more information of this test please see the video at this link: https://www.youtube.com/watch?v=v85hs8-uc1s.

VI. CONCLUSION

This paper presented an ePFC which augments the traditional PFC with the capability to use relative velocities between a drone and a target or obstacles as feedback for control. Next, the stability of the ePFC was proven using Lyapunov methods. Additionally, the presented controller was simulated and its performance relative to a tradition PFC was evaluated. The evaluation shows that the ePFC performs significantly better than a traditional PFC by reducing overshoot and settling time when navigating between waypoints. Finally, experimental results were presented which showed the actual performance of the controller.

Future work may include using an experimental system with completely on-board sensing capabilities. For a completely on-board implementation, sensing hardware would be required which would allow the drone to both localize itself in the environment as well as obstacles. Promising possibilities exist and are being utilized by various research groups (e.g., LIDAR and RGBd cameras). Potentially, the front facing camera on the ARDrone 2.0 could be used for localization using computer vision algorithms if relative position is all that is required for the application. Aside from sensing capabilities, this algorithm is very computationally inexpensive and can run on nearly any flight controller or other on-board computer, and therefore the processing requirements can be easily met with nearly any off the shelf solution today.

The proposed ePFC can be extended to a collaborative potential field control for multi-UAV for in-flight collision avoidance as well as obstacle avoidance [43]–[47] even in noisy environments where the UAV's pose is affected by localization sensors' noise [48], [49]. Cooperative sensing and learning [50]–[52] for multi-UAV can be developed based on this collaborative potential field control.

REFERENCES

[1] R. Cui, Y. Li, and W. Yan, "Mutual information-based multi-AUV path planning for scalar field sampling using multidimensional *RRT**," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 7, pp. 993–1004, Jul. 2016.
[2] S. Minaeian, J. Liu, and Y.-J. Son, "Vision-based target detection and localization via a team of cooperative UAV and UGVS," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 46, no. 7, pp. 1005–1016, Jul. 2016.
[3] J. M. de Dios, L. Merino, F. Caballero, A. Ollero, and D. Viegas, "Experimental results of automatic fire detection and monitoring with UAVs," *Forest Ecol. Manag.*, vol. 234, no. 1, p. S232, Nov. 2006.
[4] L. Merino, F. Caballero, J. R. Martínez-de Dios, J. Ferruz, and A. Ollero, "A cooperative perception system for multiple UAVS: Application to automatic detection of forest fires," *J. Field Robot.*, vol. 23, nos. 3–4, pp. 165–184, 2006.
[5] D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, and R. K. Mehra, "Forest fire monitoring with multiple small UAVs," in *Proc. Amer. Control Conf.*, vol. 5. Portland, OR, USA, Jun. 2005, pp. 3530–3535.
[6] P. B. Sujit, D. Kingston, and R. Beard, "Cooperative forest fire monitoring using multiple UAVs," in *Proc. 46th IEEE Conf. Decis. Control*, New Orleans, LA, USA, Dec. 2007, pp. 4875–4880.
[7] C. Yuan, Z. Liu, and Y. Zhang, "UAV-based forest fire detection and tracking using image processing techniques," in *Proc. Unmanned Aircraft Syst. (ICUAS)*, Denver, CO, USA, Jun. 2015, pp. 639–643.
[8] R. R. Pitre, X. R. Li, and R. Delbalzo, "UAV route planning for joint search and track missions: An information-value approach," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 3, pp. 2551–2565, Jul. 2012.
[9] A. Birk, B. Wiggerich, H. Bülow, M. Pfingsthorn, and S. Schwertfeger, "Safety, security, and rescue missions with an unmanned aerial vehicle (UAV)," *J. Intell. Robotic Syst.*, vol. 64, no. 1, pp. 57–76, 2011.
[10] T. Tomic *et al.*, "Toward a fully autonomous UAV: Research platform for indoor and outdoor urban search and rescue," *IEEE Robot. Autom. Mag.*, vol. 19, no. 3, pp. 46–56, Sep. 2012.
[11] M. A. Goodrich *et al.*, "Supporting wilderness search and rescue using a camera-equipped mini UAV," *J. Field Robot.*, vol. 25, nos. 1–2, pp. 89–110, 2008.
[12] D. Erdos, A. Erdos, and S. E. Watkins, "An experimental UAV system for search and rescue challenge," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 28, no. 5, pp. 32–37, May 2013.
[13] M. Quaritsch *et al.*, "Networked UAVs as aerial sensor network for disaster management applications," *Elektrotechnik und Informationstechnik*, vol. 127, no. 3, pp. 56–63, 2010.
[14] I. Maza, F. Caballero, J. Capitan, J. R. Martínez-de Dios, and A. Ollero, "Experimental results in multi-UAV coordination for disaster management and civil security applications," *J. Intell. Robot. Syst.*, vol. 61, nos. 1–4, pp. 563–585, 2011.

[15] P. Bupe, R. Haddad, and F. Rios-Gutierrez, "Relief and emergency communication network based on an autonomous decentralized UAV clustering network," in *Proc. SoutheastCon*, Fort Lauderdale, FL, USA, Apr. 2015, pp. 1–8.
[16] H. M. La *et al.*, "Mechatronic systems design for an autonomous robotic system for high-efficiency bridge deck inspection and evaluation," *IEEE/ASME Trans. Mechatronics*, vol. 18, no. 6, pp. 1655–1664, Dec. 2013.
[17] S. J. Mills, J. J. Ford, and L. Mejías, "Vision based control for fixed wing UAVs inspecting locally linear infrastructure using skid-to-turn maneuvers," *J. Intell. Robot. Syst.*, vol. 61, nos. 1–4, pp. 29–42, 2011.
[18] Z. Li, Y. Liu, R. Walker, R. Hayward, and J. Zhang, "Towards automatic power line detection for a UAV surveillance system using pulse coupled neural filter and an improved hough transform," *Mach. Vis. Appl.*, vol. 21, no. 5, pp. 677–686, 2010.
[19] R. Bloss, "Unmanned vehicles while becoming smaller and smarter are addressing new applications in medical, agriculture, in addition to military and security," *Ind. Robot Int. J.*, vol. 41, no. 1, pp. 82–86, 2014.
[20] J. J. Roldan, G. Joossen, D. Sanz, J. del Cerro, and A. Barrientos, "Mini-UAV based sensory system for measuring environmental variables in greenhouses," *Sensors*, vol. 15, no. 2, pp. 3334–3350, 2015.
[21] M. Rossi *et al.*, "Gas-drone: Portable gas sensing system on UAVs for gas leakage localization," in *Proc. IEEE SENSORS*, Valencia, Spain, Nov. 2014, pp. 1431–1434.
[22] H. M. La, W. Sheng, and J. Chen, "Cooperative and active sensing in mobile sensor networks for scalar field mapping," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 1, pp. 1–12, Jan. 2015.
[23] M. Shaohua, X. Jinwu, and L. Zhangping, "Navigation of micro aerial vehicle in unknown environments," in *Proc. 25th Chin. Control Decis. Conf. (CCDC)*, Guiyang, China, 2013, pp. 322–327.
[24] S. Grzonka, G. Grisetti, and W. Burgard, "A fully autonomous indoor quadrotor," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 90–100, Feb. 2012.
[25] S. Shen, N. Michael, and V. Kumar, "Autonomous multi-floor indoor navigation with a computationally constrained MAV," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Shanghai, China, May 2011, pp. 20–25.
[26] I. Sa and P. Corke, "System identification, estimation and control for a cost effective open-source quadcopter," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2012, pp. 2202–2209.
[27] A. Kendall, N. Salvapantula, and K. Stol, "On-board object tracking control of a quadcopter with monocular vision," in *Proc. Int. Conf. Unmanned Aircraft Syst. (ICUAS)*, St. Paul, MN, USA, May 2014, pp. 404–411.
[28] S. Shen, N. Michael, and V. Kumar, "Autonomous indoor 3D exploration with a micro-aerial vehicle," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, St. Paul, MN, USA, 2012, pp. 9–15.
[29] J.-E. Gomez-Balderas, P. Castillo, J. A. Guerrero, and R. Lozano, "Vision based tracking for a quadrotor using vanishing points," *J. Intell. Robot. Syst.*, vol. 65, nos. 1–4, pp. 361–371, 2012.
[30] L. Meier *et al.*, "PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision," *Auton. Robots*, vol. 33, nos. 1–2, pp. 21–39, 2012.
[31] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Shanghai, China, May 2011, pp. 2520–2525.
[32] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed," *IEEE Robot. Autom. Mag.*, vol. 17, no. 3, pp. 56–65, Sep. 2010.
[33] A. Dogan, "Probabilistic approach in path planning for UAVs," in *Proc. IEEE Int. Symp. Intell. Control.*, Oct. 2003, pp. 608–613.
[34] Y. Kuwata, T. Schouwenaars, A. Richards, and J. How, "Robust constrained receding horizon control for trajectory planning," in *Proc. AIAA Guid. Navig. Control Conf.*, 2005, pp. 1–12.
[35] A. A. Neto, D. G. Macharet, and M. F. M. Campos, "Feasible path planning for fixed-wing UAVs using seventh order Bézier curves," *J. Brazil. Comput. Soc.*, vol. 19, no. 2, pp. 193–203, 2013.
[36] A. Altmann, M. Niendorf, M. Bednar, and R. Reichel, "Improved 3D interpolation-based path planning for a fixed-wing unmanned aircraft," *J. Intell. Robot. Syst.*, vol. 76, no. 1, pp. 185–197, 2014.
[37] H. M. La, R. S. Lim, W. Sheng, and J. Chen, "Cooperative flocking and learning in multi-robot systems for predator avoidance," in *Proc. IEEE 3rd Annu. Int. Conf. Cyber Technol. Autom. Control Intell. Syst. (CYBER)*, Nanjing, China, May 2013, pp. 337–342.
[38] H. M. La and W. Sheng, "Multi-agent motion control in cluttered and noisy environments," *J. Commun.*, vol. 8, no. 1, pp. 32–46, 2013.
[39] H. M. La and W. Sheng, "Dynamic target tracking and observing in a mobile sensor network," *Robot. Auton. Syst.*, vol. 60, no. 7, pp. 996–1009, 2012.
[40] D. E. Sanabria. *Ardrone Simulink Development Kit V1.1*. Accessed on Dec. 10, 2014. [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/43719-ar-drone-simulink-development-kit-v1-1
[41] *Motion Analysis Systems*. Accessed on Jan. 14, 2015. [Online]. Available: http://www.motionanalysis.com
[42] A. C. Woods. *Dynamic Target Tracking With Ardrone*. Accessed on May 10, 2015. [Online]. Available: https://youtu.be/v85hs8-uc1s
[43] H. M. La, T. Nguyen, T. D. Le, and M. Jafari, "Formation control and obstacle avoidance of multiple rectangular agents with limited communication ranges," *IEEE Trans. Control Netw. Syst.*, to be published, doi: 10.1109/TCNS.2016.2542978.
[44] H. M. La and W. Sheng, "Adaptive flocking control for dynamic target tracking in mobile sensor networks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, St. Louis, MO, USA, Oct. 2009, pp. 4843–4848.
[45] H. M. La, T. H. Nguyen, C. H. Nguyen, and H. N. Nguyen, "Optimal flocking control for a mobile sensor network based a moving target tracking," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Oct. 2009, pp. 4801–4806.
[46] H. M. La, R. S. Lim, W. Sheng, and H. Chen, "Decentralized flocking control with a minority of informed agents," in *Proc. 6th IEEE Conf. Ind. Electron. Appl.*, San Antonio, TX, USA, Jun. 2011, pp. 1851–1856.
[47] T. Nguyen, T.-T. Han, and H. M. La, "Distributed flocking control of mobile robots by bounded feedback," in *Proc. 54th Annu. Allerton Conf. Commun. Control Comput. (Allerton)*, Monticello, IL, USA, Sep. 2016, pp. 563–568.
[48] H. M. La and W. Sheng, "Flocking control of multiple agents in noisy environments," in *Proc. IEEE Int. Conf. Robot. Autom.*, Anchorage, AK, USA, May 2010, pp. 4964–4969.
[49] A. D. Dang, H. M. La, and J. Horn, "Distributed formation control for autonomous robots following desired shapes in noisy environment," in *Proc. IEEE Int. Conf. Multisensor Fusion Integr. Intell. Syst. (MFI)*, Baden-Baden, Germany, Sep. 2016, pp. 285–290.
[50] H. M. La, R. Lim, and W. Sheng, "Multirobot cooperative learning for predator avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 52–63, Jan. 2015.
[51] H. M. La, W. Sheng, and J. Chen, "Cooperative and active sensing in mobile sensor networks for scalar field mapping," in *Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE)*, Madison, WI, USA, Aug. 2013, pp. 831–836.
[52] H. M. La and W. Sheng, "Distributed sensor fusion for scalar field mapping using mobile sensor networks," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 766–778, Apr. 2013.

Alexander C. Woods received the B.S. and M.S. degrees in mechanical engineering from the University of Nevada at Reno, Reno, NV, USA, in 2013 and 2016, respectively.

From 2014 to 2016, he was with the Advanced Robotics and Automation Laboratory, University of Nevada. He is currently an Engineer with Nevada Nanotech Systems Inc., Sparks, NV, USA. His current research interests include dynamic systems and control, robotics, and unmanned autonomous systems.

Hung M. La (M'09–SM'14) received the B.S. and M.S. degrees in electrical engineering from the Thai Nguyen University of Technology, Thai Nguyen, Vietnam, in 2001 and 2003, respectively, and the Ph.D. degree in electrical and computer engineering from Oklahoma State University, Stillwater, OK, USA, in 2011.

He is the Director of the Advanced Robotics and Automation Laboratory, and an Assistant Professor with the Department of Computer Science and Engineering, University of Nevada at Reno, Reno, NV, USA. From 2011 to 2014, he was a Post-Doctoral Research Fellow and then a Research Faculty Member with the Center for Advanced Infrastructure and Transportation, Rutgers University, Piscataway, NJ, USA.

Dr. La is an Associate Editor of the IEEE TRANSACTIONS ON HUMAN–MACHINE SYSTEMS, and a Guest Editor of the *International Journal of Robust and Nonlinear Control*.