Proceedings of the 2016 IEEE International Symposium on Safety, Security and Rescue Robotics, EPFL, Lausanne, Switzerland, October 23-27, 2016

Study of Improved Pilot Performance using Automatic Collision Avoidance for Tele-operated Unmanned Aerial Vehicles

Daman Bareiss Jur van den Berg Jake J. Abbott Kam K. Leang

***Abstract*— This paper studies the application of automatic collision avoidance algorithms to help pilots improve their maneuvering of unmanned aerial vehicles (UAVs). Automatic collision avoidance technology can help reduce the cognitive workload of a pilot, especially when flying UAVs through cluttered and complex unstructured environments. The feedforward-based algorithm reviewed herein exploits the dynamics of the aerial robot and if a collision is predicted, the algorithm modifies the operator's input to avoid a collision. The algorithm has recently been implemented on a quadcopter UAV with on-board computation and sensing. To quantify the improvement in pilot performance compared to other methods, human-subject studies were conducted using a simulated quadcopter UAV running the collision avoidance algorithms. Specifically, a comparison is made between the feedforward-based algorithm, the basic risk field algorithm (a variant on potential field), and full manual control. Experimental results show that the feedforward-based algorithm performs *significantly* better than manual control by lowering the number of collisions and increasing the UAV's average speed, both of which are extremely vital, for example, for UAV-assisted search-and-rescue applications. Compared to the potential-field based algorithm, the feedforward algorithm enabled the pilot to operate the UAV with significantly higher average speeds without drastically affecting the number of collisions.**

I. Introduction

Unmanned aerial vehicles (UAVs), particularly small low-cost platforms, have gained considerable attention for civil and commercial applications ranging from mapping [1], precision farming [2], traffic management [3], and environmental monitoring [4]. More recently, the emergence of small multirotor UAVs (such as quadcopters), which can access indoor locations and maneuver through environments that are hard to reach or unsafe for humans, has captured the attention of the public-safety sector and law-enforcement officials as a viable tool to enhance situational awareness for search and rescue, law enforcement, and/or emergency response [5]–[7]. However, one of the most daunting tasks for even a skilled UAV pilot is controlling the aircraft for collision avoidance, especially in tight and compact environments such as inside of a partially collapsed building where usually the only feedback information is a live-camera feed through first-person view (FPV) mode (see Fig. 1 illustrating the typical UAV system that is controlled through a remote command station). Thus, automatic collision-avoidance technology for tele-operated UAVs (as well as mobile ground robots) is critical and necessary to allow pilots to focus on higher-priority tasks such as locating survivors and acting quickly to help assist survivors or call for additional support.

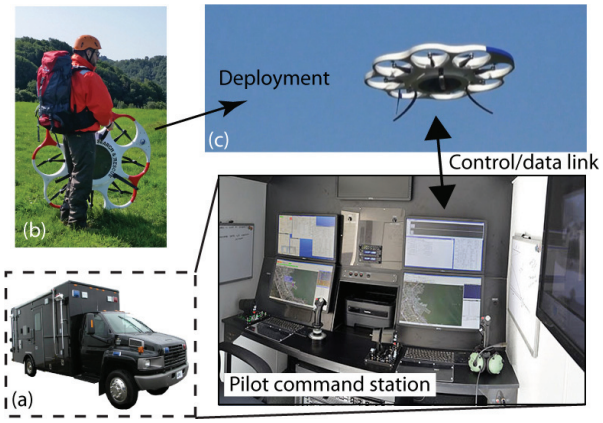

Fig. 1: A UAV system for search and rescue and emergency response, where pilots control the unmanned aerial vehicle (UAV) through (a) a mobile command station or similar interface following (b) deployment of (c) the UAV with on-board cameras and sensors. Control signals and data flow between the UAV and command station. Images courtesy of Mike Richards and Drone America, Inc.

To quantitatively investigate the impact of automatic collision avoidance technology on UAV-pilot performance, the contribution of this paper is a human-subject study that compares the performance between a feedforward-based collision avoidance algorithm [8], [9], a basic risk (potential) field algorithm [10], and full manual control. Specifically, experiments are described where pilots operate a simulated UAV system running the algorithms through three maze-like environments. In the experiments, the number of collisions, the path length, trial time, and average speed are recorded. There are four hypotheses being tested in this paper. First, it is hypothesized that the feedforward-based algorithm in this paper will result in fewer collisions than manual control. Second, of the trials that do not collide, it is hypothesized that there will be higher operating speeds (i.e., shorter completion times) with the feedforward-based algorithm over manual control. Third, it is hypothesized that the feedforward-based algorithm will result in fewer collisions than the potential-field variant, the basic risk field algorithn [10]. Lastly, it is hypothesized that the feedforward-based algorithm will provide higher operating speeds than the basic risk field algorithm. The first, second, and fourth hypotheses are supported by

D. Bareiss and K. K. Leang are with the Design, Automation, Robotics & Control (DARC) Lab, Dept. of Mechanical Engineering at the Univ. of Utah, E-mails: {daman.bareiss, kam.k.leang}@utah.edu; J. van den Berg is with the School of Computing at the Univ. of Utah and J. J. Abbott is with the Dept. of Mechanical Engineering at the Univ. of Utah. The authors are also members of the Univ. of Utah Robotics Center.

This material is based upon work supported by the National Science Foundation, Partnership for Innovation Program, Grant No. 1430328.

978-1-5090-4349-1/16/$31.00 ©2016 IEEE

the experiments, while the third hypothesis is inconclusive, but suggests that there is not a significant difference in the frequency of collisions between the two algorithms.

II. State-of-the-Art in Collision Avoidance

Early research on motion planning and collision avoidance for mobile robots include potential-field planners [11] and the vector field histogram (VFH) approach [12]. Improvements were made to the VFH in [13]. Although these algorithms are effective, they do have some potential limitations for applications such as search and rescue. For instance, these algorithms have an inherent requirement to keep the robot some minimum distance away from the obstacles in the environment. The need to maintain a minimum distance from obstacles such as walls comes from the fact that the algorithms do not explicitly consider the dynamics of the robot. However, in a search and rescue scenario the robot may need to be controlled near walls in constrained environments or in order to quickly to survey the environment. Thus, factoring in the robot's dynamics can improve the performance of the collision-avoidance process.

Collision avoidance methods that do consider the robot's dynamics typically require a global knowledge of the environment [14]–[16]. Unfortunately, such information may not be readily available at the time of search and rescue and often, not practical. Furthermore, these algorithms are more computationally expensive than the reactive planners such as potential-fields and VFH.

Herein, a feedforward-based local collision-avoidance algorithm is presented that has similarities to both classes of collision avoidance algorithms [8], [9]. More specifically, the algorithm considers the robot's dynamics and extrapolates the robot's trajectory given an operator's input. The resulting trajectory is checked for collisions against the obstacles in the environment. If a collision is predicted, the user's input is modified to guide the robot along a collision-free trajectory. Similar to the reactive planners, such as potential-field, this algorithm only requires a limited knowledge of the local environment in the immediate vicinity of the robot. It also has similarities to more complex planners through the propagation of the trajectory using the robot's dynamics. However, rather than optimizing this trajectory explicitly, the algorithm is designed to alter the user's input directly that results in collision-free motion while maintaining the user's intent as closely as possible.

The remainder of this paper is structured as follows. The feedforward-based automatic collision-avoidance algorithm is reviewed in Sec. III. The methodology of the experiments is presented in Sec. V and the results are presented and discussed in Sec. VI. Finally, concluding remarks and a discussion of future work are presented in Sec. VIII.

III. Automatic Collision Avoidance

This section provides a review of the feedforward-based automatic collision avoidance (ACA) algorithm studied in this paper. The full theoretical details for the deterministic and stochastic approaches are presented in [8] and [9], respectively.

TABLE I: Quadcopter model parameters

Parameter	k_{pv}	k_{px}	k_{dx}	k_{py}	k_{dy}	k_{pz}	k_{drag}
Value	10.0	150.0	2.5	150.0	2.5	3.5	0.25

A. System Equations and Robot Workspace

Consider a robot with general, nonlinear equations of motion and a state space of arbitrary dimension m. Let $\mathcal{X} \subset \mathbb{R}^m$ be the state space of the robot and let $\mathcal{U} \subset \mathbb{R}^n$ be the control input space. The continuous-time equations of motion of the robot are defined by the function $\mathbf{f} \in \mathcal{X} \times \mathcal{U} \to \mathbb{R}^m$, i.e.,

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{1}$$

where $\mathbf{x}(t) \in \mathcal{X}$ and $\mathbf{u}(t) \in \mathcal{U}$ are the state and control input at time t, respectively.

Given an initial state $\mathbf{x} = \mathbf{x}(0)$ and a constant control input $\mathbf{u}$ up to the time-horizon τ, the state of the robot for $t > 0$ is defined by

$$\mathbf{x}(t) = \mathbf{g}(\mathbf{x}, \mathbf{u}, t), \tag{2}$$

where $\mathbf{g} \in \mathcal{X} \times \mathcal{U} \times \mathbb{R} \to \mathcal{X}$ represents the solution to the differential equation (1).

The workspace in which the robot maneuvers is $\mathbb{R}^d$, where typically $d \leq 3$. The obstacles occupy a subset of the workspace, hence $\mathcal{O} \subset \mathbb{R}^d$. Let $\mathcal{R}(\mathbf{x}) \subset \mathbb{R}^d$ denote the subset of the workspace occupied by the robot when it is in state $\mathbf{x} \in \mathcal{X}$. Then, a colliding state is defined as $\mathcal{R}(\mathbf{x}(t)) \cap \mathcal{O} \neq \emptyset$.

Remark: *In order to maintain compatibility with the implementation of on-board sensing, those regions of the workspace that are occluded by the obstacles as seen from the current state of the robot are also considered obstacles. In other words, the subspace of the workspace that cannot be seen by the robot is also an obstacle and belongs in $\mathcal{O}$.*

The system model has a 12-dimensional state $\mathbf{x} = [\mathbf{p}^T, \mathbf{v}^T, \mathbf{r}, \mathbf{w}]^T \in \mathcal{X}$ that consists of position $\mathbf{p} \in \mathbb{R}^3$, velocity $\mathbf{v} \in \mathbb{R}^3$, Euler angles $\mathbf{r} \in \mathbb{R}^3$, and angular velocity $\mathbf{w} \in \mathbb{R}^3$. The 4-dimensional control input $\mathbf{u} = [r_x^\star, r_y^\star, v_z^\star, w_z^\star]^T \in \mathcal{U}$ consists of the desired roll and pitch angles, $r_x^\star$ and $r_y^\star$, respectively, the desired vertical velocity $v_z^\star$, and the desired yaw rate $w_z^\star$. Assuming a quadcopter UAV system, the equations of motion are

$$\dot{\mathbf{p}} = \mathbf{v}, \tag{3}$$

$$\dot{\mathbf{v}} = R\left[0, 0, k_{\text{pv}}(v_z^\star - v_z)\right]^T - \mathbf{g} - k_{\text{drag}}\mathbf{v}, \tag{4}$$

$$\dot{\mathbf{r}} = \mathbf{w}, \tag{5}$$

$$\dot{\mathbf{w}} = \begin{bmatrix} k_{\text{px}}(r_x^\star - r_x) - k_{\text{dx}} w_x \\ k_{\text{py}}(r_y^\star - r_y) - k_{\text{dy}} w_y \\ k_{\text{pz}}(w_z^\star - w_z) \end{bmatrix}, \tag{6}$$

where R is the rotation from the quadcopter body frame into the world frame, the terms $k_{\text{pv}}, k_{\text{px}}, k_{\text{dx}}, k_{\text{py}}, k_{\text{dy}}, k_{\text{pz}}$, and k_{drag} are gains whose values (given in Table I) are equal to an analogous physical system. Similar to many physical quadcopters, the simulated quadcopter has a maximum limit on the roll and pitch angles of 0.35 rad (approximately 20.0 degrees).

B. Problem Statement

The collision avoidance problem is to find a minimal change $\Delta\mathbf{u} \in \mathcal{U}$ to the control input $\mathbf{u} \in \mathcal{U}$ given the initial state $\mathbf{x} \in \mathcal{X}$ of the robot that avoids collisions with obstacles within a time horizon τ, hence

$$\begin{aligned} &\text{minimize: } \Delta\mathbf{u}^T Q \Delta\mathbf{u}, \\ &\text{subject to: } \forall t \in [0,\tau] :: \mathcal{R}(\mathbf{g}(\mathbf{x}, \mathbf{u}+\Delta\mathbf{u}, t)) \cap \mathcal{O} = \emptyset, \end{aligned} \tag{7}$$

where $Q \in \mathbb{R}^{n \times n}$ is a positive-definite weighting matrix.

C. Approach

Given the robot's current state $\mathbf{x}$ and the current control input $\mathbf{u}$ (from the operator), the positions of the robot in the future are found by

$$\mathbf{p}(t, \Delta\mathbf{u}) \approx \mathbf{p}^\star(t) + J(t)\Delta\mathbf{u}, \tag{8}$$

where $\mathbf{p}^\star(t)$ is the position the robot would obtain if the operator's input remains constant, i.e., $\Delta\mathbf{u} = \mathbf{0}$, and $J(t)$ is the Jacobian of the position with respect to the input.

For a trajectory that is determined to be collision free ($\forall t \in [0,\tau] :: \mathcal{R}(\mathbf{p}^\star(t)) \cap \mathcal{O} = \emptyset$), the operator's current input $\mathbf{u}$ is deemed safe and does not need to be changed, hence the change in input is set to zero, i.e., $\Delta\mathbf{u} = \mathbf{0}$. Conversely, if a collision does occur ($\forall t \in [0,\tau] :: \mathcal{R}(\mathbf{p}^\star(t)) \cap \mathcal{O} \neq \emptyset$) the operator's input leads to a collision and must be corrected in order for the robot to obtain a collision-free trajectory, hence the change in input is non-zero, i.e., $\Delta\mathbf{u} \neq \mathbf{0}$. The process to select a non-zero change in input is decribed next.

Let $\mathbf{p}_c$ be the first point along the trajectory in which the robot collides with an obstacle (see Fig. 2), thus

$$\mathbf{p}_c = \mathbf{p}^\star(\min\{t \in [0,\tau] \mid \mathcal{R}(\mathbf{p}^\star(t)) \cap \mathcal{O} = \emptyset\}). \tag{9}$$

Given a unit normal vector $\mathbf{n}$ of the obstacle $\mathcal{O}$ that points into the free workspace, consider a halfspace with the same normal $\mathbf{n}$ (pointing toward the free space) that provides a convex approximation of the local free space. The halfspace is located at the collision point $\mathbf{p}_c$, determined by Eq. (9).

From Eq. (9), a linear constraint is defined on the position $\mathbf{p}(\tau, \Delta\mathbf{u})$ of the robot at time τ, hence

$$\mathbf{n}^T \mathbf{p}(\tau, \Delta\mathbf{u}) > \mathbf{n}^T \mathbf{p}_c. \tag{10}$$

The constraint on the robot's position in Eq. (10) can be transformed into a constraint on its change in input $\Delta\mathbf{u}$ by substituting in Eq. (8), thus

$$\mathbf{n}^T J(\tau)\Delta\mathbf{u} > \mathbf{n}^T(\mathbf{p}_c - \mathbf{p}^\star(\tau)). \tag{11}$$

Finally, Eq. (7) is approximated using Eq. (11) as

$$\begin{aligned} &\text{minimize: } \Delta\mathbf{u}^T Q \Delta\mathbf{u} \\ &\text{subject to: } \mathbf{n}^T J(\tau)\Delta\mathbf{u} > \mathbf{n}^T(\mathbf{p}_c - \mathbf{p}^\star(\tau)), \end{aligned} \tag{12}$$

where solving this convex optimization, such as is done by the RVO library in [17], provides a collision-free change in input $\Delta\mathbf{u}$. In summary, the total control input provided to the robot is $\mathbf{u} + \Delta\mathbf{u}$. Additional details of the automatic collision avoidance algorithm can be found in [9].

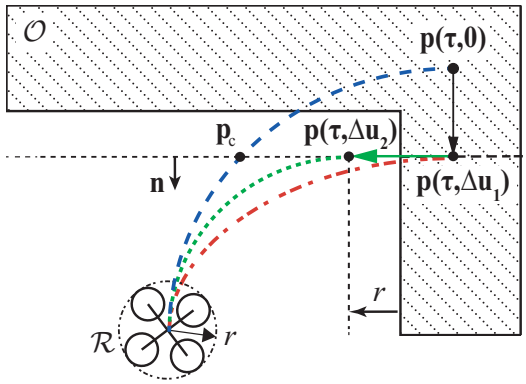

Fig. 2: Robot with its bounding geometry $\mathcal{R}$, where the estimated trajectory for a user's input is given with the desired position at the time horizon $\mathbf{p}(\tau, 0)$ causing a collision with the obstacle $\mathcal{O}$ at $\mathbf{p}_c$. The new collision-free input at time τ is solved for; however, in convex corners this process needs to be iterated to detect possible collisions between the updated trajectory. Given the first iteration, the new estimated trajectory with the desired position as $\mathbf{p}(\tau, \Delta\mathbf{u}_1)$ and a new input is chosen if a collision occurs. This is repeated until the trajectory is collision free as shown by $\mathbf{p}(\tau, \Delta\mathbf{u}_2)$.

D. Handling Convex Corners and Edges through Iteration

The use of an approximation of a convex region of the local free space near the robot's trajectory means that it cannot be assumed that the newly selected control input $\mathbf{u} + \Delta\mathbf{u}$ avoids collisions with respect to *all* obstacles for all time $t \in [0,\tau]$. This is true near convex edges or corners of the workspace as shown in Fig. 2. However, the approach can simply be repeated in an iterative fashion to solve this problem, where additional details can be found in [8].

E. Including Yaw as an Additional Degree-of-Freedom

In Refs. [8], [9], yaw motion is treated as a redundant degree-of-freedom and is held constant. However, to better enable a pilot to survey an area in a search-and-rescue scenario, holding the yaw constant limits performance. Yaw motion is especially necessary if the pilot is flying through a first-person video feed using a forward-facing camera.

To enable the yaw to be controlled by the pilot, the yaw degree-of-freedom is not affected by the algorithm. The algorithm determines the feedforward trajectory estimate assuming the user's current commanded yaw rate remains constant over the time horizon τ, similar to the desired roll and pitch angles. Through the Jacobian in Eq. (8), the algorithm can determine new roll and pitch angles and vertical velocity to avoid a collision given a constant yaw rate.

IV. Potential-Field for UAV Tele-operation

The potential-field algorithm provides a repulsive force on the robot based on the distance between the robot and an obstacle detected by a range sensor [11]. The repulsive force increases as distance between the robot and the obstacle decreases. However, since this algorithm produces forces as a function of distance, it can still provide large repulsive forces even if the robot is moving *away* from a nearby obstacle, which is not desirable for tele-operated applications.

In [18], the potential-field algorithm was augmented to include the velocity and acceleration constraints of a robot. However, in this approach the repulsive force is zero when the robot has no velocity component towards the obstacle. This is undesirable for a tele-operation application where a user may suddenly provide a control input towards that obstacle and the robot can collide if a repulsive force cannot counter that input fast enough. Lam et al. [10] presented the basic risk field (BRF), a potential-field variant, to address this concern. Their approach provides a small repulsive force for nearby obstacles even when there is no velocity component of the robot towards that obstacle. Therefore, there is always a small repulsion force pushing the robot away from obstacles, but the repulsive force is only a large force when reacting to velocities toward the given obstacle. The potential function $P(d, v_i)$ that defines these repulsive forces given in [10] is

$$t_{res}(d, v_i) = \frac{a_{max} v_i}{2 d a_{max} - v_i^2}, \tag{13}$$

$$P(d, v_i) = \begin{cases} 1, & \text{if } t_{res} \leq 0 \\ 1, & \frac{1}{t_{res}(d,v_i)} + \frac{1}{d} \geq \frac{1}{G} \\ G\left(\frac{1}{t_{res}(d,v_i)} + \frac{1}{d}\right), & \text{otherwise,} \end{cases}$$

where a_{max} is the maximum acceleration away from an obstacle, v_i is the robot's velocity component towards the obstacle, d is the distance between the robot and obstacle, and G is a gain to tune the magnitude of the repulsive gain.

V. Experimental Methods

A. Subjects

Three experiments were performed by twenty four subjects (eight per experiment). The subjects were recruited from the University of Utah student population. The subjects had the physical ability to use a commercial video game console controller and were at least eighteen years of age. The subjects were not compensated for their participation. The experiment was approved by the University of Utah Institutional Review Board (IRB No. 00082430).

B. Device

The quadcopter model provided in Sec. III-A is implemented in simulation on a desktop computer with an Intel Core i5-3470 3.2 GHz processor, 8GB RAM, and 64-bit Ubuntu 12.04 operating system. The algorithms are implemented using the Robot Operating System (ROS) [19]. The aircraft is flown in first-person view (FPV) mode, where simulated FPV is made available to pilots operating the UAV. The simulator includes a 2D LIDAR to detect the obstacles in the environment in real-time during the experiments. Figure 4 shows a test pilot operating the simulator.

Three environments were created in simulation using the V-REP software package [20] as shown in Fig. 3. The environments each used the same starting position of the quadcopter but have different finish locations. The corridors of each environment are either 1 m, 1.5 m, or 2 m wide. The simulated quadcopter has a diameter of 0.564 m.

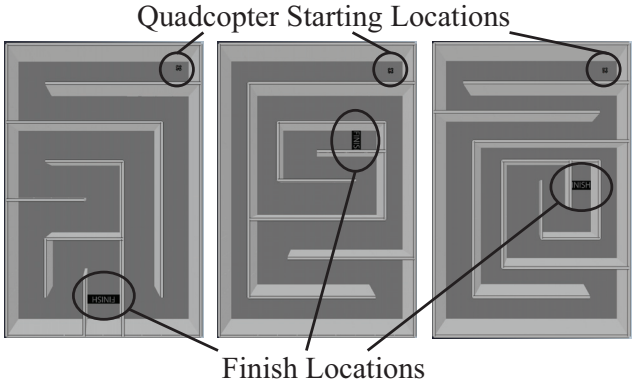

Fig. 3: Three different mazes used during the experimental trials. The mazes have the same starting location, but different finish locations as annotated in the image.

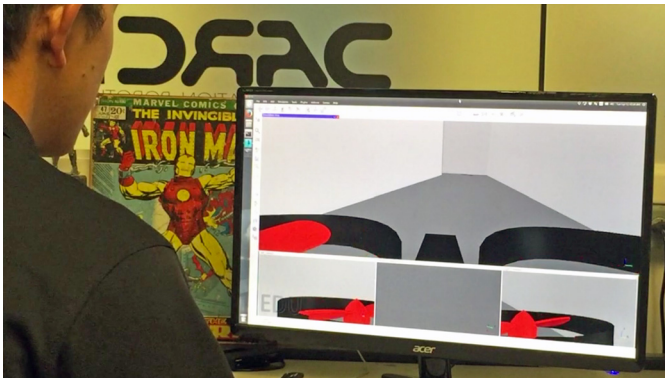

Fig. 4: Photograph of test pilot operating the simulator.

C. Design

A full-factorial repeated-measures design is used for the three experiments. There are two factors being considered in the experiments: the control method (manual, automatic collision avoidance, or basic risk field) and the environment (mazes shown in Fig. 3). A block design is used in which the three environments are presented to the participant in eight blocks of three, for twenty four total trials. The order of the mazes in each block of three mazes is a random permutation. Each environment is seen an equal number of times by all participants.

The experiments compare pairs of control methods. The first and second experiments compare manual control to the ACA algorithm. These studies test the hypotheses that the ACA algorithm will result in fewer collisions than manual control, and that when collisions do not occur, the ACA algorithm will enable higher operating speeds. During a pilot study, it was observed that many subjects preferred to fly the quadcopter similar to a car, where they provided a constant forward input and steered the UAV through yaw. However, the yaw rate input being applied with maximum roll and/or pitch can lead to collisions due to the assumptions in the ACA algorithm's development. Thus, the first experiment allowed the quadcopter to yaw, which led to a relatively high number of collisions. The second experiment is designed such that the quadcopter cannot yaw. Instead, the camera rotates and the pilot's roll and pitch commands are defined in the camera

frame and mapped into the robot frame. The third experiment compares the BRF algorithm to the ACA algorithm. This experiment tests the hypotheses that the ACA algorithm will result in fewer collisions than the BRF algorithm, and that the ACA algorithm will enable higher operating speeds than the BRF algorithm. In [10], the simulated quadcopter was a velocity controlled robot with simplified dynamics. The simulations in this paper, however, utilize the full nonlinear dynamics of the quadcopter with inputs including roll and pitch, i.e., accelerations. There were unnecessary oscillations observed when using the BRF so a damping term was included for the roll and pitch inputs.

In each experiment, each participant completed half of their trials (twelve trials) using one of the two control methods and then the second half with the alternate control method. The experiments alternated the order of the control methods for each successive participant to attempt to minimize learning effects on the results. For example, the first subject would be tested first using manual control and second with the ACA algorithm, then the second subject would first use the ACA algorithm and use manual control second.

D. Procedure

For each experiment, the subjects completed two sessions with at least 24 hrs between each session. For each session, the subject sat at a desk and held a wireless game console controller while directly facing a 24-in desktop computer monitor located approximately 24 in away from the subject. The subject is instructed that, for each trial, they should attempt to complete each maze as fast as possible while avoiding collisions. A collision is indicated by the screen turning red and the current trial stopping automatically.

Before each session, the subject is required to practice with the control method of that session for three minutes. The quadcopter model is the same during the practice as it is during the experiments. The environment during the three minutes of practice consists of 2 m wide hallways and has a similar appearance to the experiment's environments.

During the experiment, the subject knows which control method they are using but not the technical details of the algorithm (or lack thereof). A verbal cue is given to the subject before each trial began. A trial ends automatically with a collision or the crossing of the finish line, after which the process is repeated for all twelve trials of that session. Each session typically lasts 15–30 min for each subject, with each subject completing two sessions.

In the first experiment, the subjects are informed that the automatic collision avoidance algorithm could still have collisions if a full roll or pitch input is applied at the same time as a yaw input, but no further information about the algorithm is provided. In the second and third experiments, when the quadcopter cannot yaw, the subjects are not provided with this additional information about the algorithm.

E. Measures

To quantify the performance of the subjects operating the simulated quadcopter, their performance is defined by several metrics: if a collision occurred (yes/no), the time to complete each trial, the path length traveled, and the average operating speed. The time, path length, and average operating speed are recorded for the duration of each trial, which ended in either a collision or crossing the finish line.

The collision data can be represented as a binomial distribution and analyzed using the Friedman test [21]. The maps had a small effect on the results and are distributed equally in the experiment design, therefore, the collisions are only categorized on one level: by which collision-avoidance algorithm (or lack thereof) is being used by the subject. The remaining measures can be analyzed using a two-way ANOVA [22].

VI. Experimental Results

The experimental results (the means, standard deviations, and comparison metrics) are summarized in Table II. Analysis of operating speeds for *only* the trials that are completed (i.e., no collisions) is provided in Table III. Figure 5 provides box plots of the data set for measures in which ANOVA is performed (i.e., all measures except collisions). Table IV shows if each participant's individual results supported (✓) or contradicted (✕) the hypothesis being tested with statistical significance at 95% confidence. A "-" represents that no conclusions can be drawn for or against the hypothesis with statistical significance. For hypothesis two, that the ACA algorithm enables higher operation speeds than manual control for *completed* trials, a value of "N/A" means that no manual trials were completed, or in other words *all* twelve trials resulted in a collision for that participant and no statistical testing can be completed regarding the second hypothesis.

A. Experiment One: Automatic Collision Avoidance (ACA) vs. Manual Control With Yaw

In regards to Hypothesis 1, the statistical results that include the data from all subjects and all trials (Table II) indicate that the ACA algorithm results in significantly less collisions than manual control. The mean number of collisions decreased by 40% from manual control when using the ACA algorithm. When considering the results for individual subjects (Table IV(a)), four out of the eight subjects showed a significant improvement when using ACA and the other four subjects are inconclusive. No subject showed significant improvement when using manual control.

In regard to Hypothesis 2, the statistical results from all subjects and *completed* trials (Table III) indicate that the ACA algorithm enabled pilots to fly the UAV with higher average operating speeds compared to manual control with a 70% increase in speed. Considering only the individual results (Table IV(a)), two of the eight subjects showed improvement in their operating speeds with statistical significance. Another two subjects are inconclusive, with no statistical significance between their speeds. The remaining four subjects did not complete a single trial with manual control. Although the statistics of their operating speeds cannot be assessed, the fact that they did not complete a single trial manually demonstrates the usefulness of the ACA algorithm. No subject showed improvement when using manual control compared to the ACA algorithm.

TABLE II: Distribution statistics for all subjects and all trials

(a) Sample Means (μ) and Standard Deviations (s)

		Collisions		Time		Path Len.		Avg. Speed	
		μ	s	μ	s	μ	s	μ	s
E1	Man.	0.89	0.32	66	74	32	25	0.64	0.27
	ACA	0.53	0.501	74	56	54	25	0.87	0.27
E2	Man.	0.86	0.34	42	38	29	23	0.83	0.35
	ACA	0.10	0.31	76	21	71	11	0.97	0.12
E3	BRF	0.073	0.26	112	34	76	17	0.70	0.092
	ACA	0.10	0.31	77	17	73	10	0.96	0.12

(b) Comparison Metrics

	Collisions		Time		Path Len.		Avg. Speed	
	Chi-Sq.	p	F	p	F	p	F	p
E1	33	1.2e-8	1.5	0.23	43	1.1e-9	55	1.6e-11
E2	103	3.7e-24	68	1.9e-13	351	1.7e-38	15	1.8e-4
E3	0.58	0.44	136	6.7e-22	4.1	0.045	468	1.4e-44

TABLE III: Distribution statistics for completed trials only, for all subjects

(a) Sample Means (μ) and Standard Deviations (s)

		Avg. Speed	
		μ	s
E1	Man.	0.46	0.14
	ACA	0.78	0.23
E2	Man.	0.76	0.18
	ACA	0.96	0.12
E3	BRF.	0.70	0.09
	ACA	0.95	0.11

(b) Comparison metrics

	Avg. Speed	
	F	p
E1	21	2.5e-5
E2	26	1.7e-6
E3	268	5.1e-37

B. Experiment Two: ACA vs. Manual Control Without Yaw

The second experiment addressed Hypotheses 1 and 2 as well, but, unlike the first experiment, in this experiment the quadcopter cannot yaw. Hypothesis 1 is strongly supported by the statistical results for all subjects and all trials (Table II). There is an observed 88% decrease in the number of collisions from manual control when using the ACA algorithm in this experiment. Looking at the individual subject's results in Table IV(b) shows that, in fact, *every* subject had fewer collisions with the ACA algorithm than they did with manual control with statistical significance.

Considering the statistical results from the completed trials for all subjects (Table III), Hypothesis 2 is supported as well, where the subjects had a higher mean average operating speed with statistical significance with a 26% increase in speed. The individual results in Table IV(b) supports Hypothesis 2 as well. Four of the eight subjects showed improved operating speeds with statistical significance. One of the eight subjects is inconclusive and the remaining three could not be analyzed because they did not complete a single trial using manual control. None of these subjects showed improvement with significance when using manual control.

C. Experment Three: ACA vs. Basic Risk Field (BRF)

This experiment addressed Hypotheses 3 and 4. Regarding Hypothesis 3, there is no conclusive evidence found from the experiment. There is no significant difference in the mean number of collisions between the ACA algorithm and the BRF algorithm. This is the case for all subjects and all trials (Table II) as well as each individual subject (Table IV).

Although Hypothesis 3 is inconclusive, the results of this experiment strongly support Hypothesis 4. The subjects performed at higher operating speeds with the ACA algorithm compared to the BRF algorithm as seen by the statistics for all subjects and all trials (Table II) and the completed trials only (Table III). When looking at the individual subject's result in Table IV(c), the hypothesis is supported by *all* eight subjects independently.

TABLE IV: Hypothesis results for individual subjects

(a) Exp. 1

	H1	H2
S1	✓	N/A
S2	-	-
S3	-	N/A
S4	-	✓
S5	✓	N/A
S6	✓	N/A
S7	✓	✓
S8	-	-

(b) Exp. 2

	H1	H2
S1	✓	✓
S2	✓	N/A
S3	✓	N/A
S4	✓	N/A
S5	✓	✓
S6	✓	✓
S7	✓	✓
S8	✓	-

(c) Exp. 3

	H3	H4
S1	-	✓
S2	-	✓
S3	-	✓
S4	-	✓
S5	-	✓
S6	-	✓
S7	-	✓
S8	-	✓

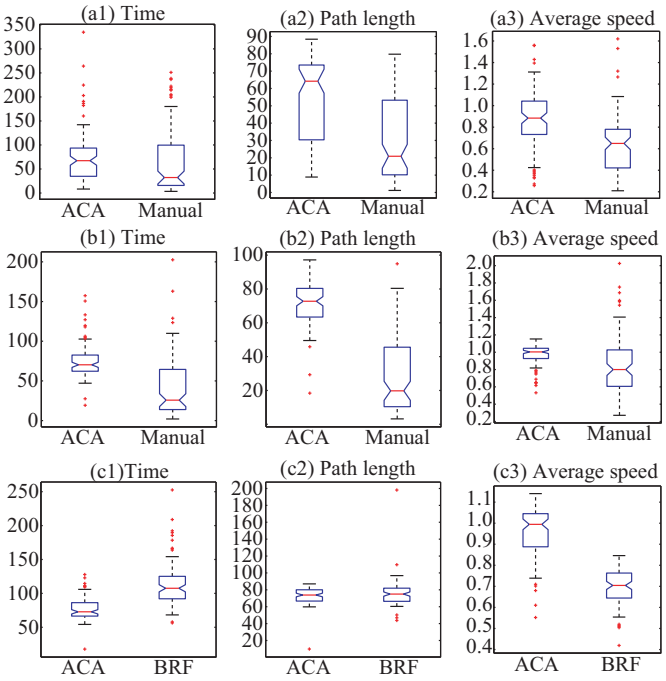

Fig. 5: Box plots for the experiments comparing automatic collision avoidance (ACA) algorithm, manual control (Manual), and basic risk field (BRF) algorithm. Results in (a1)–(a3) show ACA versus manual control of the UAV with yaw. Results in (b1)–(b3) show ACA versus manual control of the UAV without yaw. Results in (c1)–(c3) show performance of ACA versus BRF algorithm. Left column shows the trial time [(a1)–(c1)], middle column shows the path length [(a2)–(c2)], and right column shows the average speed [(a3)–(c3)].

VII. Discussion

It is observed that there is a large difference in the speed increase between the first and second experiment, both comparing ACA to manual control. It is hypothesized that this resulted from the fact that the subjects were informed during the first experiment that a large yaw input at the same time as roll and pitch could result in collision with the ACA algorithm and tended to fly differently in both the manual and ACA trials, typically flying a short distance and stopping and

then rotating in place.

The third hypothesis is that there would be fewer collisions using the ACA algorithm than the BRF algorithm. The results of the third experiment are inconclusive with regard to this hypothesis. It was expected that the BRF would perform well regarding collisions due to the potential field's conservative nature, leading to the slower operation speed, so these results are not surprising. The ACA algorithm could be made more conservative to reduce its number of collisions as well. Given the mean number of collisions from the third experiment, obtaining statistical significance would require a much higher power of the study through an increased number of subjects. However, the effect size is small (0.12 when considering the BRF as the control group) and a study with higher power is not likely necessary from a practical standpoint.

The third experiment supported the fourth hypothesis, which is that the ACA algorithm would perform at higher average operating speeds than the BRF algorithm. An increase in speed of approximately 37% is observed in the experiment. This improvement in performance is predicted due to the ACA algorithm's ability to adjust the input based on the full dynamics and, for example, allowing the robot to strafe along a wall and only alter the trajectory when a collision is predicted. The BRF is more conservative with a repulsive force always applied with a magnitude based on velocity and distance to the wall. In the current implementation, the forces were tuned to be able to travel through the narrow corridors, which are problematic for potential-field algorithms [23], while still being able to decelerate the robot to a stop when it approaches a wall at high speed. Although it is predicted that the BRF could allow higher operating speeds in less-constrained environments, in applications like search-and-rescue a tightly constrained environment can be expected.

VIII. Conclusions and Future Work

This paper studied UAV-pilot performance with and without the assistance of collision-avoidance algorithms. A comparison was done between a feedforward-based collision-avoidance algorithm, a basic risk field (potential field-based) algorithm, and full manual control. Human-subject tests were performed where pilots operated a simulated UAV system running the algorithms through three maze-like environments. In the experiments, the number of collisions, the path length, trial time, and average operating speed were recorded. The experimental results showed that the proposed feedforward-based automatic collision-avoidance algorithm is capable of significantly improving a pilot's performance compared to manual control and the basic risk field algorithm for the tele-operation of UAVs.

Further studies would include more extensive comparison of the feedforward-based collision avoidance algorithm to other local collision avoidance methods and field studies of the algorithms on various UAV platforms including fixed-wing configurations.

References

[1] F. Nex and F. Remondino, "UAV for 3D mapping applications: a review," *Applied Geomatics*, vol. 6, no. 1, pp. 1 – 15, 2013.

[2] C. Zhang and J. M. Kovas, "The application of small unmanned aerial systems for precision agriculture: a review," *Precision Agriculture*, vol. 13, no. 6, pp. 693–712, 2012.

[3] P. S. Lin, L. Hagen, K. Valavanis, and H. Zhou, "Vision of unmanned aerial vehicle (UAV) based traffic management for incidents and emergencies," in *12th World Congress on Intelligent Transport Systems*, pp. 1–12, 2005.

[4] D. Hausamann, W. Zirnig, G. Schreier, and P. Strobl, "Monitoring of gas pipelines a civil UAV application," *Aircraft Engineering and Aerospace Technology*, vol. 77, no. 5, pp. 352 – 360, 2005.

[5] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey, "Supporting wilderness search and rescue using a cameraequipped mini UAV," *Journal of Field Robotics*, vol. 25, no. 1-2, pp. 89 – 110, 2008.

[6] M. Kumar, K. Cohen, and B. Homchaudhuri, "Cooperative control of multiple uninhabited aerial vehicles for monitoring and fighting wildfires," *Journal of Aerospace Computing, Information, and Communication*, vol. 8, no. 1, pp. 1–16, 2011.

[7] S. Waharte and N. Trigoni, "Supporting search and rescue operations with UAVs," in *IEEE Int. Conf. on Emerging Security Technologies*, pp. 142–147, 2010.

[8] J. Israelsen, M. Beall, D. Bareiss, D. Stuart, E. Keeney, and J. van den Berg, "Automatic collision avoidance for manually tele-operated unmanned aerial vehicles," in *IEEE Int. Conf. on Robotics and Automation*, pp. 6638–6643, 2014.

[9] D. Bareiss, J. van den Berg, and K. K. Leang, "Stochastic automatic collision avoidance for tele-operated unmanned aerial vehicles," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 4818–4825, 2015.

[10] T. M. Lam, H. W. Boschloo, M. Mulder, and M. M. van Paassen, "Artificial force field for haptic feedback in UAV teleoperation," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 39, no. 6, pp. 1316–130, 2009.

[11] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[12] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 19, pp. 1179–1187, 1989.

[13] I. Ulrich and J. Borenstein, "VFH*: Local obstacle avoidance with look-ahead verification," in *IEEE Int. Conf. on Robotics and Automation*, pp. 2505–2511, 2000.

[14] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in *IEEE Int. Conf. on Robotics and Automation*, pp. 4569–4574, 2011.

[15] A. Bry and N. Roy, "Rapidly exploring random belief trees for motion planning under uncertainty," in *IEEE Int. Conf. on Robotics and Automation*, pp. 723–730, 2011.

[16] S. Patil, J. van den Berg, and R. Alterovitz, "Estimating probability of collision for safe planning under gaussian motion and sensing uncertainty," in *IEEE Int. Conf. on Robotics and Automation*, pp. 3238–3244, 2012.

[17] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Proc. Int. Symp. on Robotics Research*, pp. 3–19, 2011.

[18] J. Minguez, L. Montano, and O. Khatib, "Reactive collision avoidance for navigation with dynamic constraints," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 588–594, 2002.

[19] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "Ros: An open-source robot operating system," in *IEEE Int. Conf. on Robotics and Automation: Workshop on Open-Source Software*, p. 5, 2009.

[20] E. Rohmer, S. Singh, and M. Freese, "V-REP: A versatile and scalable robot simulation framework," in *IEEE Int. Conf. on Intelligent Robots and Systems*, pp. 1321–1326, 2013.

[21] M. Hollander and D. A. Wolfe, *Nonparametric Statistics*. 1973.

[22] J. Neter, M. Kutner, W. Wasserman, and C. Nachtsheim, *Applied Linear Statistical Models*. McGraw-Hill/Irwin, 1996.

[23] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation," in *IEEE Int. Conf. on Robotics and Automation*, 1991.