Haptic Collision Avoidance for a Remotely Operated Quadrotor UAV in Indoor Environments

Adam M. Brandt and Mark B. Colton
Department of Mechanical Engineering
Brigham Young University
Provo, UT, United States
abrandt8@gmail.com, colton@byu.edu

***Abstract*—Quadrotors are well-suited for indoor flight because of their ability to hover and maneuver in confined spaces. However, remote operation of these vehicles is challenging because of the loss of sensory perception. This paper presents methods for using force feedback exerted by the command input device on the hand of the pilot to assist in avoiding collisions while navigating in a 3D indoor environment. Three algorithms for calculating the feedback forces are presented and tested in simulation-based user studies. The force feedback and control system is demonstrated by piloting a physical quadrotor in an indoor environment.**

***Keywords*—unmanned aerial vehicles, haptics, force feedback.**

I. INTRODUCTION

Quadrotors are a type of rotor-craft that employ four coplanar rotors to generate thrust for hovering and maneuvering. The rotors are typically positioned symmetrically about a central body, which houses the battery, autopilot and other computing hardware, as shown in Fig. 1. Quadrotors have two primary advantages over fixed-wing aircraft for certain unmanned aerial vehicle (UAV) applications. First, they have the ability to hover, and second, they are omni-directional, allowing them to maneuver freely in any direction. These capabilities of quadrotors make them a popular choice for indoor flight.

The use of UAVs in indoor environments has several potential applications, including search and rescue in mines or dangerous buildings, monitoring and assessment of buildings or factories after toxic spills, inspection of inaccessible regions in buildings, and military and law enforcement applications. These applications may also be useful in outdoor environments, such as crowded urban settings. Operating UAVs remotely is desirable in these applications for many reasons. First, the vehicle can be smaller, lighter, more discreet, and less expensive because it does not require a human occupant. Second, when the vehicle must pass through dangerous or hazardous areas, there is no possibility for harm to the operator.

One of the most important problems with the remote operation of flying vehicles is that sensory perception is lost because the human operator is not physically collocated with the remote vehicle that he or she is piloting. In many cases designers attempt to ameliorate the sensory deficit by equipping the vehicle with an onboard camera to give the remote pilot a forward view of the environment in which it operates. Although such cameras are useful, the visibility from a remote camera has been described as "looking at the world through a soda straw" [1]. This decrease in situational awareness may result in increased time to complete a task, increased workload on the human operator, decreased mission effectiveness, or collisions with obstacles within the environment.

The problem of decreased situational awareness is especially acute with quadrotors. Although the ability to move in any direction provides a considerable advantage over fixed-wing UAVs, omni-directional motion also enables a quadrotor to move in directions that are not within the camera's field of view. This makes the quadrotor susceptible to collisions in lateral, vertical, and rear directions. The limited payload capabilities of quadrotors typically makes the inclusion of additional cameras (and the more sophisticated hardware for processing and transmission of the video data) impractical.

The purpose of this paper is to explore one possible solution to this problem: the use of force feedback to give the pilot haptic information about the quadrotor's environment. This feedback is used to supplement the limited visual feedback using another natural and highly sensitive sensory channel: the sense of touch. In the present work, the primary purpose of haptic feedback is to assist the pilot in avoiding obstacles. When the quadrotor is approaching an obstacle in any direction, the input device exerts appropriate forces on the pilot's hand to help him or her avoid a collision. The approach taken in this work is for the feedback forces to be directed away from the obstacle, to inform the pilot of the quadrotor's motion toward an obstacle and to encourage a change of

Figure 1. A typical quadrotor with four coplanar rotors around the main body. This quadrotor does not have a camera.

978-1-4244-6588-0/10/$25.00 ©2010 IEEE

motion in the direction of the applied force. The force is not designed to overpower the user, but to serve as a warning to assist in avoiding the obstacle.

A. Related Work

Most research on haptic collision avoidance has been related to unmanned ground vehicles (UGVs), which are remotely controlled or autonomous mobile robots. Borenstein and Koren [11] used a vector field histogram (VFH) method to compute forces to help the user avoid collisions, while at the same time guiding the vehicle toward a target. They found that the VHF method was robust and allowed fast movement of the robot without it stopping in front of obstacles. Similarly, Lee et al. [12] performed a user study with ground robots using sensors to find range information and convert the range information into forces displayed to the user through a haptic device. Their user study found that the haptic feedback increased operator performance and reduced collisions without significantly increasing navigation time.

Hong et al. [13] demonstrated the use of haptic feedback with a robot capable of climbing stairs. The haptic feedback not only assisted in collision avoidance, but also in stair climbing as the forces helped the user to feel when a leg would touch the ground. Barnes and Counsell [14] explored haptic feedback collision avoidance, with the user having full control of the ground robot as well as the robot having semi-autonomous control. The results of the study showed that operator performance improved when haptic information was given when the user had full control, but it was inconclusive in the semi-autonomous control mode.

UAVs differ from UGVs in certain key areas. First, UAVs have a greater number of degrees of freedom, thus requiring more directions of control and introducing a greater possibility for collisions. Second, the payload capabilities of UAVs are much lower, and the amount of equipment and computational power is correspondingly decreased. Much of the previous work in haptic collision avoidance for UAVs has been done by Lam, Mulder, and van Paassen at the Delft University of Technology in the Netherlands. Their earlier research focused on development of artificial risk fields used to calculate the forces felt by the operator when approaching potential hazards. More recently they have explored the effects of transmission time delays [2] and degraded visual interfaces [3].

Boschloo et al. [4] explored the use of two types of risk fields with two types of force vector calculations applied to each for determining the appropriate haptic feedback forces. They tested a basic risk field, in which the feedback forces increase exponentially as the vehicle approaches an obstacle, and a parametric risk field, in which the feedback forces increase according to a more complex function of the vehicle's distance from an obstacle. In the latter, the distance is divided into distinct zones, with the feedback forces changing based on which zone the UAV is currently in relative to the wall. Both approaches were explored using radial and normal risk vectors. The simulation of these methods was done in two dimensions, without hardware or humans in the loop. They found that the parametric risk field with radial risk vectors produced the fewest collisions in the simulations.

This research was continued by Lam et al. [5], who performed a user study on the two risk fields. In this simulation the user had a three dimensional view of the world, but motion was limited to translation in two dimensions. The force feedback was provided by an electro-hydraulic side-stick and the simulation was done in a fixed-base part-task simulator. The results of the experiment were quantified by the number of collisions, elapsed time, average speed, workload using NASA-TLX [6], and a user questionnaire. The experiment showed that the haptic feedback did help to avoid collisions, but sometimes required a higher workload when the repulsive forces from the haptic control device required counteracting forces from the operator.

B. Overview

This paper presents the development of three force feedback algorithms for use in piloting quadrotor UAVs in indoor environments. A software environment to simulate 3D motion and 3D force feedback is described, and user studies using this environment are presented. The system is demonstrated by piloting a physical quadrotor in an indoor environment with force feedback.

II. Force Feedback Algorithms

Force feedback has the potential to help pilots avoid collisions while piloting remote UAVs, but if the methods for calculating feedback forces are not properly developed, the forces may actually be detrimental in terms of difficulty in piloting the aircraft, causing collisions, increased mental workload, etc. Three candidate force feedback algorithms were developed for comparison against each other and against no force feedback. This section describes the algorithms, which include time-to-impact (TTI), dynamic parametric field (DPF), and virtual spring (VS).

A. Time-to-Impact (TTI)

The time-to-impact (TTI) algorithm is inspired by the work by others, such as Balas [7], who have used inverse time-to-collision to assist automobile drivers in maintaining safe distances while following other automobiles. As the TTI decreases, the haptic force increases to indicate that a collision is becoming increasingly likely. The TTI is calculated using

$$\mathrm{TTI} = \frac{d}{v}, \tag{1}$$

where v is the current velocity of the UAV in the direction of the obstacle, and d is the current distance to the obstacle. The feedback force F is inversely proportional to the time to impact,

$$F = \frac{k}{\mathrm{TTI}} = \frac{kv}{d}, \tag{II}$$

where k is a tunable parameter to scale the feedback forces. It is selected based on the maximum force output of the haptic interface, experimentally determined force ranges for maximum effectiveness, and user comfort. The feedback force based on the TTI method is shown as a function of distance and velocity in Fig. 2. In this figure the force is normalized by the maximum exertable force of the haptic interface. A plateau is visible for distances and velocities that result in saturation of the haptic interface force capabilities.

B. Dynamic Parametric Field (DPF)

With the dynamic parametric field (DPF) algorithm, as the quadrotor approaches an obstacle it passes through four zones that are used to determine the forces applied to the pilot's hand, as shown in Fig. 3. In this figure, d is the distance of the quadrotor from the wall, d_{max} is the distance from the wall at which the interface begins to exert feedback forces on the pilot, d_{stop} is a nominal required stopping distance for a given velocity, d_{mid} is the beginning of the transition zone, and d_{min} is the minimum required stopping distance for a given approach velocity. The various zones are described as follows:

- *Safe Zone*. When the quadrotor is in the safe zone, at a distance greater than d_{max}, no feedback force is exerted on the pilot; it is assumed that the quadrotor is not in danger of colliding with the obstacle because of the large distance and/or small closing speed.
- *Warning Zone*. When $d_{mid} < d < d_{max}$, the force increases from 0% at d_{max} to 60% of the maximum force at d_{mid}. The distance d_{stop} is the distance required to stop if the quadrotor decelerates at one half of the maximum pitch angle. The distance at which feedback is initially applied (d_{max}) is equal to d_{stop} plus a constant offset. This constant was added so that even when the quadrotor has little or no velocity, there will still be a buffer between it and the obstacle. Otherwise, if the quadrotor were near the wall but with little closing velocity, the quadrotor could slowly drift into the wall with little or no feedback force applied to warn the pilot.
- *Transition Zone*. At d_{mid} the force jumps to 80% of the maximum force. When $d_{min} < d < d_{mid}$, the force transitions linearly from 80% to 100% of the maximum force. The distance d_{mid} is proportional to d_{min}.
- *Collision Zone*. When $d < d_{min}$ a collision with the obstacle is unavoidable because of the distance from the wall and the speed of the quadrotor. The value of d_{min} is determined from the dynamic model of the quadrotor; at a given approach velocity v the minimum distance required to stop is calculated at the maximum pitch angle of the quadrotor, because quadrotors accelerate and decelerate by changing pitch angle. In this zone the maximum feedback force is applied in an attempt to minimize the collision velocity.

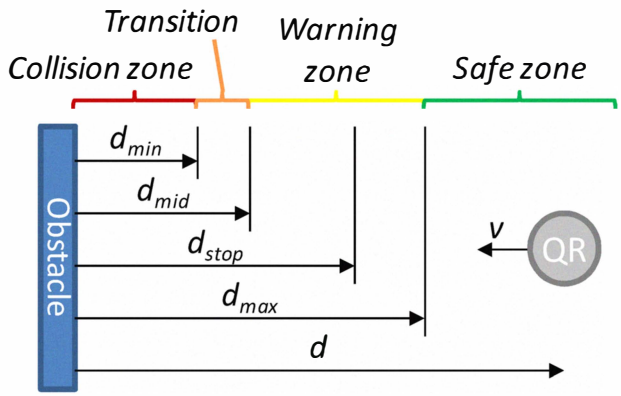

Figure 3. Force zones for the dynamic parametric field (DPF) algorithm. As the quadrotor (QR) approaches an obstacle, it passes through the warning zone, transition zone, and collision zone (when a collision is unavoidable, based on the dynamics of the quadrotor).

The force output with respect to velocity and distance is shown in Fig. 4. The salient feature of this approach is that the force field is based on the dynamics of the quadrotor, namely its deceleration capabilities.

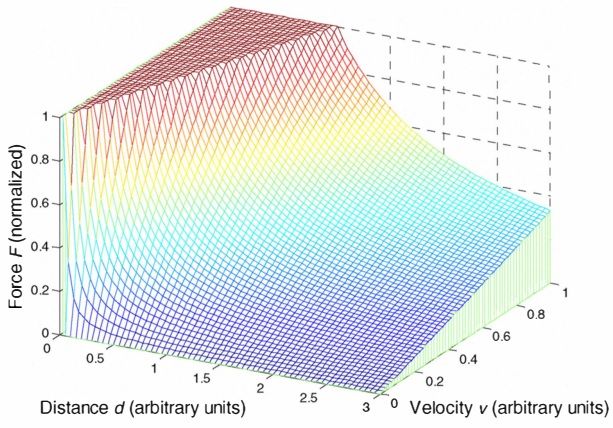

Figure 2. Force vs. velocity and distance from obstacle for the time-to-impact (TTI) algorithm. The distance and velocity are presented in arbitrary units and the force is normalized by the maximum force of the haptic interface.

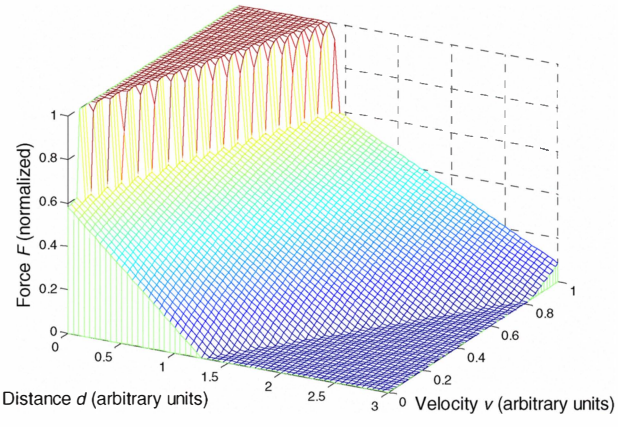

Figure 4. Force vs. velocity and distance from obstacle for the dynamic parametric field (DPF) algorithm. The distance and velocity are presented in arbitrary units and the force is normalized by the maximum force of the haptic interface.

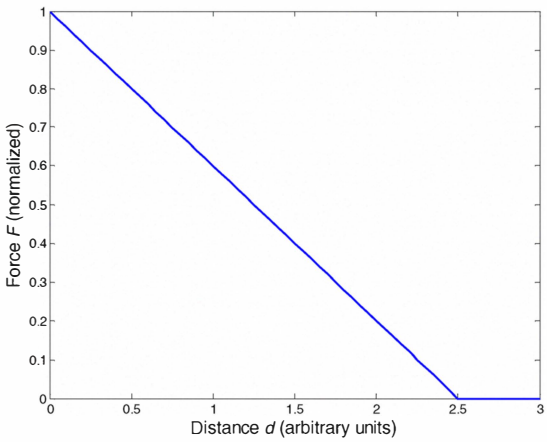

Figure 5. Force vs. distance from obstacle for the spring algorithm. The distance is presented in arbitrary units and the force is normalized by the maximum force of the haptic interface. For this example, a force deadband (no exerted force) exists for $d > 2.5$ units.

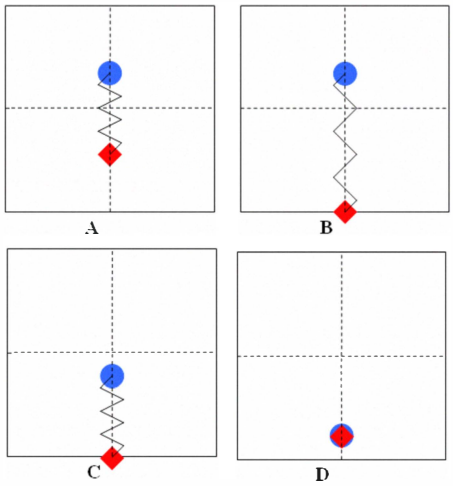

Figure 6. Example of input position dependency. The blue circles represent the position of the haptic input device. The red diamonds represent the desired input position, as calculated from the force feedback algorithm. Each box represents the x-y plane of the input device workspace.

C. *Virtual Spring (VS)*

The virtual spring algorithm is based on the quadrotor's distance from obstacles, whereas the previous algorithms are functions of distance and velocity. In this algorithm, a virtual spring is connected between the quadrotor and the obstacle. As the quadrotor approaches the obstacle, the force increases linearly from zero at the outer limit of the spring region (which may be at the center of a hallway, for example) to the maximum force at the obstacle as shown in Fig. 5. The spring region begins a distance d_{spring} away from the obstacle, and the force increases linearly according to

$$F = F_{\max} \frac{d_{spring} - d}{d_{spring}}. \tag{3}$$

D. *Input Device Position Dependency*

The force feedback algorithms described in the previous sections apply forces that are a function of position and velocity of the quadrotor only. Because of the dynamics of the quadrotor, by the time the feedback forces subside after warning the pilot of an obstacle, the quadrotor is accelerating away from the obstacle, resulting in an unintended velocity in this direction. In some cases, where obstacles are close together, the forces cause instabilities and collisions. To address this issue, another dependency was added to each of the algorithms: the position of the input device.

In each algorithm, the output force is scaled between zero and one and multiplied by the maximum force. With input position dependency, instead of multiplying this output by the maximum force, it is multiplied by a maximum input device position value, to get a desired input device position $p_{desired}$. In the present work, a haptic interface with a ±50 mm range is used. To illustrate the position dependency, the basic spring algorithm results in a desired input device position given by

$$p_{desired} = 50 \frac{d_{spring} - d}{d_{spring}} \tag{4}$$

To calculate the feedback force, the actual position p of the input device is compared to the desired position and divided by 100, which is the maximum possible difference in positions. This scales the output between zero and one so it can be multiplied by the maximum force to get the force output to the haptic interface:

$$F = F_{\max} \frac{p - p_{desired}}{100} \tag{5}$$

This is analogous to connecting a virtual spring to the input device with the zero position of the spring changing according to the output from the force feedback algorithm. It rewards the user for doing the correct action: if the user moves the input device in the direction necessary to avoid a collision, the force decreases. However, if the user does not move the input in the correct direction the forces increase.

Fig. 6 illustrates the input position dependency. In Fig. 6(A), the input device position (blue circle) commands forward motion and the quadrotor is near an obstacle to the front, causing the feedback algorithm to give a desired position (red diamond) behind the actual position. The spring between the two represents the feedback force pulling the input device to the desired position. In (B) the input device has not been moved and the quadrotor has traveled closer to the obstacle. The desired position has moved further back and the effective spring force pulling the input device has increased. In (C) the input device has been pulled back closer to the desired position and the force is decreased. In (D) the position of the input device is the same as the desired position and there is no force pulling the input device in either direction.

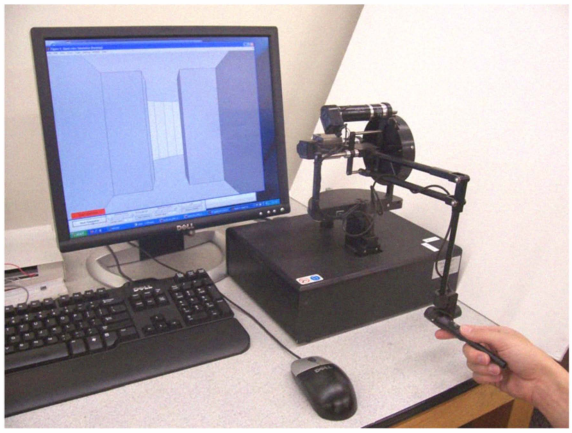

Figure 7. PHANTOM haptic interface and simulated indoor flight environment.

E. *Vertical Haptic Feedback*

One of the key contributions of this work is the inclusion of vertical (*z*-direction) haptic feedback to avoid floor and ceiling obstacles. The *z*-direction feedback was implemented using the three algorithms described previously.

III. Simulation

A simulation environment was built using Simulink, a graphical simulation environment within MATLAB. Simulink was used to model the quadrotor dynamics and the autopilot functions as described in [8], and the SensAble PHANTOM [9] haptic interface was used to command the simulated quadrotor and provide force feedback to the user. In this simulation the quadrotor is commanded in angle control mode, in which the lateral displacement of the PHANTOM commands a desired roll angle and longitudinal displacement commands a desired pitch angle. This control mode was selected because it corresponds to the actual control mode use in the physical quadrotor that will be used in subsequent experiments. A vertical displacement of the PHANTOM commands a desired vertical velocity and a rotational displacement commands a desired yaw angle rate. A graphical display was also created in MATLAB to show the view from a camera on the simulated quadrotor. The haptic interface and simulated indoor flight environment are shown in Fig. 7.

The commands generated by the user using the PHANTOM are read into a Simulink model that simulates the quadrotor dynamics and autopilot, calculates feedback forces based on the algorithms described previously, and generates the graphics for a simulated indoor environment. The highest level of the Simulink model is shown in Fig. 8. The PHANTOM commands are read in the Force/Plotting block. These commands are passed to the Autopilot function where commands to the individual quadrotor motors are generated. The Dynamics block simulates the complex quadrotor equations of motion in response to the motor commands. The states of the quadrotor are sent to the Force/Plotting block, where feedback forces are computed based on the force feedback algorithms and sent to the PHANTOM. The other blocks shown in Fig. 8 are for handling input parameters, data collection, and timing.

The simulation course consists of a hallway with two turns and various other obstacles, as shown in Fig. 9. The width and height of the hallway is 10 times the width of the quadrotor. The quadrotor begins the course in the middle of the hallway where it is pictured. In Zone 1, outlined by the dashed red lines, there is wind that pushes to the left. This is to see if the force feedback algorithms are robust against wind and also to move the quadrotor out of the middle of the hallway to make the doorway (Obstacle 2) more difficult to pass through. Obstacle 3 is a curved wall to test the effects of the algorithms against non-straight surfaces. The pilot must then maneuver the quadrotor over Obstacle 4 and under Obstacle 5, designed to test the inclusion of *z*-direction haptic feedback. Obstacle 6 is an area of almost total darkness to test the effect of having very little visual feedback. Obstacle 7 is a pillar placed in the middle of the hallway. After the pillar, the user goes toward the end wall and the simulation stops automatically when within a certain distance of this wall.

IV. Human Subject Experiment – Simulation

Human subject experiments were conducted to compare the effectiveness of the force feedback algorithms in flights of the

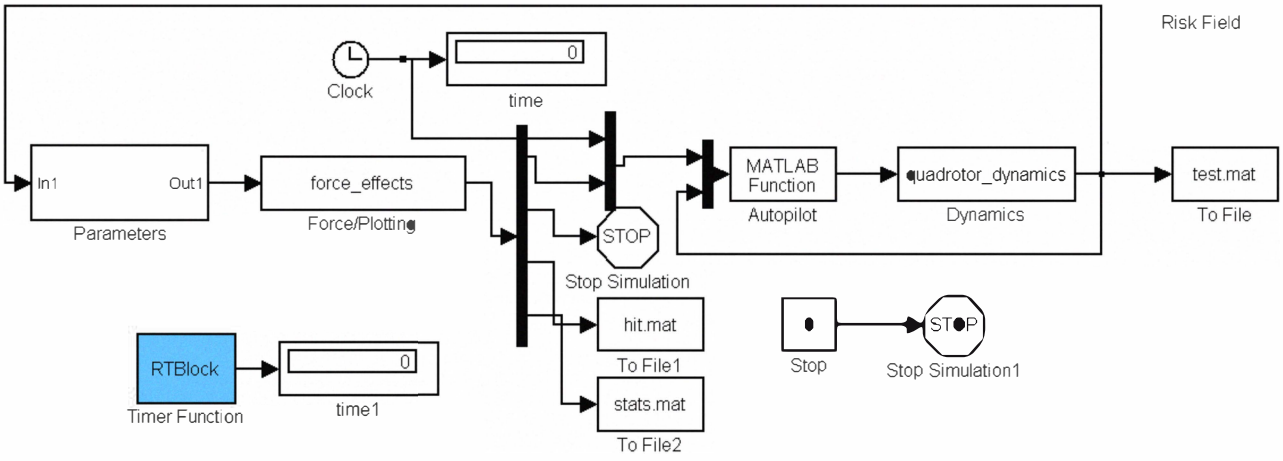

Figure 8. Top level of Simulink quadrotor simulator. This simulates the quadrotor dynamics and indoor environment and calculates feedback forces .

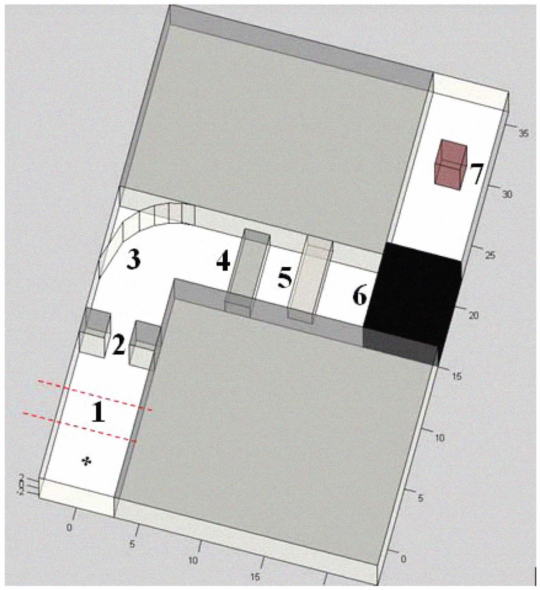

Figure 9. Overhead view of the simulated hallway course.

simulated quadrotor in the simulated indoor environment described previously. An initial study involving 15 subjects was conducted to refine the simulation and gain initial insight into the force feedback algorithms. The results of the initial study resulted in two important conclusions. First, in general higher force levels were more effective in helping to prevent collisions. This was determined by comparing each of the feedback algorithms with three different maximum force levels. Second, the three algorithms discussed in this paper were found to be the most effective of those explored in the study. The initial study compared five algorithms, including those described in this paper; two were eliminated because they did not exhibit significant improvements over the use of no force feedback.

Based on the results of the initial study, a full human subject experiment was designed to test the three force feedback algorithms in simulation against each other and also against having no force feedback. Each algorithm was tested at a single high force level, as indicated by the initial study. The experiment was designed with one factor (the force feedback algorithm) with four levels corresponding to (1) the dynamic parametric field, (2) time-to-impact, (3) virtual spring, and (4) no force feedback. Eight novice subjects each completed one block of eight runs. Each algorithm was tested twice in random order within the eight runs. As a result, each algorithm was tested a total of sixteen times. Each of the subjects was allowed two practice runs before the eight test runs. The subjects were instructed that the primary objective was to complete each test run without any collisions, and that the secondary objective was to complete the course as quickly as possible.

A. Measures

Several direct and indirect measures were chosen to assess the effectiveness of the force feedback algorithms. The direct measures indicate whether the force feedback prevented costly collisions and if it helped or hindered the pilot in efficiently completing the tasks. These include the number of collisions, the length of collisions (because it is possible to scrape along obstacles), mental workload, and time to completion. The overall workload was measured using the NASA Task Load Index (TLX) developed by Hart and Staveland [6].

The indirect measures do not tell how well a force feedback algorithm performed, but are possible indicators as to why an algorithm performed better or worse. These include the average and standard deviation of the velocity throughout the run, the path length, the average and standard deviation of the force output in the x, y, and z directions, and the average and standard deviation of the command input in the x, y, and z, directions.

B. Results

The results were analyzed using ANOVA with the block as a random variable to account for differences between users. When a measure was found to be significant, the algorithms were compared using a Tukey pairwise comparison test [10] to see which algorithms were better.

For the number of hits, the time-to-impact algorithm had significantly fewer hits than the other algorithms (including no-force). For the length of hits, the time-to-impact algorithm was significantly lower than the spring algorithm and no-force, and the spring algorithm was higher than the parametric algorithm and no-force. The top box plot in Fig. 10 shows the 95% confidence intervals for hit length for the different algorithms. No statistical difference was found for the amount of time taken to complete the course. This means that none of the algorithms decreased the time taken, but also that none of them increased the time. For workload, the VS algorithm caused a significant increase when compared to the DPF and TTI. The workload is shown in the center plot of Fig. 10.

The indirect measures that were statistically significant were the path length, the average and standard deviation of velocity, and all of the force measures. None of the command measures had significant differences. The VS algorithm caused a significant increase in the path length when compared to the TTI algorithm and no-force. For the average and standard deviation of velocity, both the DPF and TTI showed significant decreases compared to the VS algorithm and no-force. The box plot for average velocity is shown in the bottom plot of Fig. 10. The fact that the DPF algorithm and TTI both had lower average and standard deviation of velocity and performed better when comparing hits and hit length suggests that these algorithms may encourage lower velocity, which in turn may help the quadrotor to avoid obstacles.

All of the force measures were significant. Since each of the force results were similar for the separate directions, only the total (resultant) force measure will be discussed. The DPF algorithm had a lower average force than the VS algorithm and TTI, and TTI had a greater average force than the VS algorithm. It is interesting to note that although the DPF algorithm and TTI performed well in the hits measures, they had significantly different average force outputs. This suggests that it is not necessarily how much force is applied that causes fewer hits, but how the forces are applied.

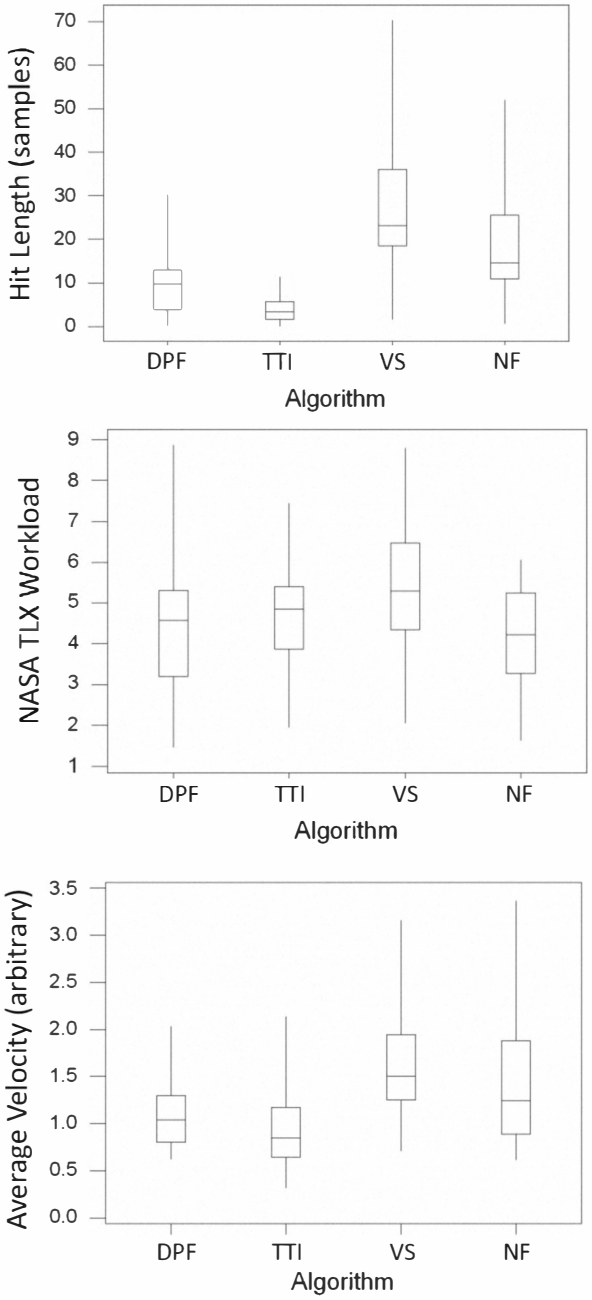

Figure 10. Sample results from human subject experiment. The boxes represent the 95% confidence intervals. The vertical lines represent the total spread of the data. DPF = dynamic parametric field, TTI = time-to-impact, VS = virtual spring, NF = no-force

V. Flight Test – Physical Quadrotor

A simple experiment was conducted to demonstrate the use of the force feedback algorithms and software in piloting an actual quadrotor UAV in an indoor environment. In this experiment, the PHANTOM was used as the input and force feedback device, and ultrasonic sensors on the quadrotor provided information on the distance of the quadrotor from obstacles.

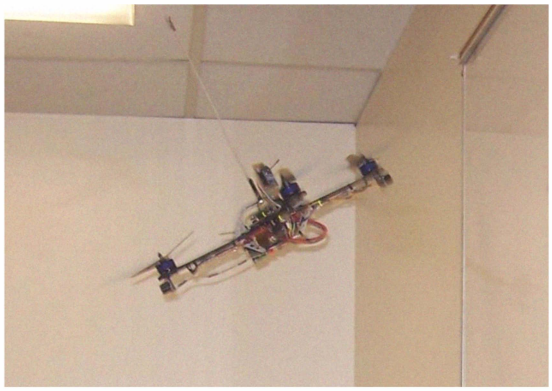

Figure 11. The quadrotor in flight in proximity of a wall under force-feedback control using the TTI algorithm.

The PHANTOM positions are read into the ground station computer and converted into commands for the quadrotor. The program on the ground station computer was written in MATLAB and communicates wirelessly with the quadrotor using an XBee radio. The program on the ground station computer uses range information, telemetry information, and PHANTOM positions to calculate feedback forces, which are then applied to the pilot via the PHANTOM.

The quadrotor used for testing is the Ascending Technologies Hummingbird (Fig. 1). The Hummingbird includes a central control unit, an autopilot, motor controllers and motors. MaxBotix LV-MaxSonar-EZ4 ultrasonic range finders were used to determine the distance from obstacles. They are capable of detecting objects at a distance of 0 to 254 inches with one-inch resolution. Three of these sensors were used for the testing: one directed out the front of the quadrotor and one directed laterally from each side.

Flight tests were performed in a rectangular room using the system described and the TTI algorithm. The tests were done with the autopilot in angle control mode. During flight, the inputs from the PHANTOM controlled the pitch and roll angles, which allowed the pilot to control forward and lateral motion. Information was sent between the computer and the quadrotor at a rate of about 28 Hz. The pilot was in the room and had a clear view of the quadrotor at all times. The quadrotor was flown for several minutes and was purposefully brought close to the walls several times to generate large feedback forces. The quadrotor in flight is shown in Fig. 11.

During each flight, the distance, velocity, and force data were recorded. Fig. 12 shows sample data for the forward direction for a portion of a flight. A negative value on the force plot corresponds to the PHANTOM generating a force in the backward direction to assist the pilot in avoiding the wall in front of the quadrotor. Because the force is dependent on both the velocity and the distance from the wall, the spikes in force output do not always occur when the quadrotor is close to a wall. In Fig. 12 at around 3600 cycles, it can be seen that the force increase is caused by the distance coming close to zero. However, around 3850 cycles, the distance is not small, but there is a large increase in velocity.

The results of the flight tests show that the pilot was able to control the quadrotor using the PHANTOM and that the forces

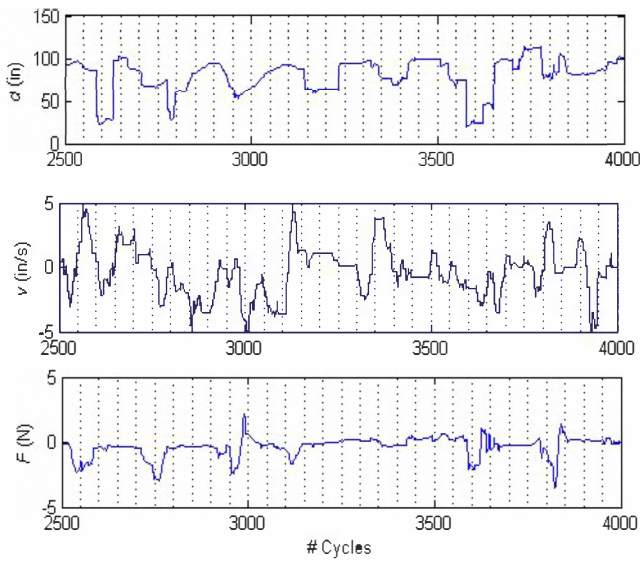

Figure 12. Sample distance to wall (top), quadrotor velocity (middle) and forward feedback force (bottom) data from flight test using the TTI algorithm.

were correctly generated as the quadrotor flew in an indoor environment. No quantitative measurements were made as to how well the forces worked, and different algorithms were not compared. The pilot noticed that appropriate forces were felt at the appropriate times, but it was unclear if the haptic feedback helped to avoid hitting the walls. The tests only showed that correct forces were generated and that the pilot could feel and understand the forces. Future work will include studies in which users pilot physical quadrotors.

VI. Conclusion

In this paper we have presented three force feedback algorithms for use in haptic control of quadrotor UAVs in indoor environments: the dynamic parametric field (DPF), time-to-impact (TTI), and virtual spring (VS) algorithms. Each algorithm was augmented with an input device position dependency to improve stability. Software was developed to simulate 3D motion and 3D force feedback in an indoor environment, and user studies were conducted using this simulation to compare the effectiveness of the three algorithms. Finally, the complete system was demonstrated on an actual quadrotor in an indoor environment.

The user study suggests that the time-to-impact algorithm is significantly better than the other algorithms and the no-force case. The TTI algorithm significantly reduced the number of collisions compared to all other algorithms without increasing the time taken to complete the task or the workload on the pilot. The initial study also indicated that higher forces are typically more effective in decreasing the number of collisions compared to lower forces for the algorithms presented in this paper.

The flight test demonstrated the feasibility of using force feedback to assist in piloting a quadrotor in an indoor environment, although no quantitative tests were conducted. When the quadrotor was flown near walls or toward walls at high velocities, appropriate forces were generated by the haptic device to suggest to the pilot to slow the quadrotor or guide it away from the walls.

References

[1] D. D. Woods, J. Tittle, M. Feil, A. Roesler, "Envisioning human-robot coordination in future operations," in *IEEE Transactions on Systems, Man and Cybernetics,* 2004, Vol. 34, pp. 210-218.

[2] T. M. Lam, M. Mulder, and M. M. Van Paassen, "Collision Avoidance in UAV Tele-operation with Time Delay," in *IEEE International Conference on Systems, Man and Cybernetics*, 7-10 October 2007, pp. 997-1002.

[3] T. M. Lam, V. D'Ameilo, M. Mulder, and M. M. Van Paassen, "UAV Tele-operation using Haptics with a Degraded Visual Interface," in *IEEE International Conference on Systems, Man and Cybernetics*, Vol. 3, 8-11 October 2006, pp. 2440-2445.

[4] H. W. Boschloo, T. M. Lam, M. Mulder, and M. M. Van Paassen, "Collision avoidance for a remotely-operated helicopter using haptic feedback," in *IEEE International Conference on Systems, Man and Cybernetics*, The Hague, The Netherlands, 10-13 October 2004, pp. 229-235.

[5] T. M. Lam, M. Mulder, and M. M. Van Paassen, "Haptic Interface for UAV collision avoidance," *International Journal of Aviation Psychology*, Vol. 17, No. 2, 2007, pp. 167-195.

[6] S.G. Hard and L.E. Staveland, "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research," in *Human Mental Workload*, P.A. Hancock and N. Meshkati, Eds., North-Holland, Amsterdam, The Netherlands, 1988, pp 139 – 183.

[7] V.E. Balas, M.M. Balas, "Driver Assisting by Inverse Time to Collision," in *Automation* Congress*, 2006. WAC '06. World,* 24-26 July 2006, pp.1-6.

[8] R. W. Beard, "Quadrotor Dynamics and Control," Brigham Young University, Tech. Rep., 2008. [Online]. Available: https://contentdm.lib.byu.edu/cgibin/showfile.exe?CISOROOT=/EER&CISOPTR=217&filename=218.pdf#search=%22Beard%22. [Accessed : Aug. 5, 2009]

[9] "Specifications Comparison for the PHANTOM Premium 1.0, 1.5, 1.5 High Force and 3.0 Haptic Devices," Jan. 7, 2009. [Online]. Available: http://www.sensable.com/documents/documents/STI_Jan2009_PremiumModelsComparison_print.pdf. [Accessed: Aug. 5, 2009].

[10] "SAS/STAT 9.2 Users Guide," 2008. pp 2516-2517, [Online]. Available:http://support.sas.com/documentation/cdl/en/statug/59654/PDF/default/statug.pdf. [Accessed: Aug. 7, 2009].

[11] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots in cluttered environments," in *IEEE International Conference on Robotics and Automation*, 13-18 May 1990, pp.572-577.

[12] S. Lee, G.S. Sukhatme, G.J. Kim, and C. Park, "Haptic control of a mobile robot: a user study," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002, vol.3, pp. 2867-2874.

[13] S. Hong, J. Lee, and S. Kim, "Generating artificial force for feedback control of teleoperated mobile robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1999, vol.3, pp.1721-1726.

[14] D. P. Barnes and M. S. Counsell, "Haptic Communication for Remote Mobile Manipulator Robot Operations," in *Proc. of 81h American Nuclear Society Topical Meeting on Robotics & Remote Systems*, April 1999.