Semi-Autonomous Human-UAV Interfaces for Fixed-Wing Mini-UAVs

Morgan Quigley
Brigham Young University
Provo, UT, USA
mquigley@byu.edu

Michael A. Goodrich
Computer Science Department
Brigham Young University
Provo, UT, USA
mike@cs.byu.edu

Randal W. Beard
Elec. and Computer Eng. Dept.
Brigham Young University
Provo, UT, USA
beard@ee.byu.edu

***Abstract*— We present several human-robot interfaces that support real-time control of a small semi-autonomous UAV. These interfaces are designed for searching tasks and other missions that typically do not have a precise predetermined flight plan. We present a detailed analysis of a PDA interface and describe how our other interfaces relate to this analysis. We then offer quantative and qualitative performance comparisons of the interfaces, as well as an analysis of their possible real-world applications.**

I. Introduction

The semi-autonomous fixed-wing mini-UAV is an emerging class of vehicle that will have significant importance to many fields, with applications including short-range military reconnaissance, rural search-and-rescue, law enforcement, and any other task that would benefit from a small, low-cost aerial vehicle [1] [2]. Because small and lightweight mini-UAVs are quite difficult to fly manually, a successful high-level interface could dramatically increase the number of potential users and applications of these aircraft.

Previous work has produced several prototype mini-UAVs which have significant autopilot capabilities [3]. On each UAV, the on-board autopilot offers several control modes that represent various levels of autonomy. The highest level of autonomy has the ability to automatically take off, follow a mutable set of GPS waypoints, and automatically land. However, this paper seeks to explore the capabilities of semi-autonomous control modes – where the user supplies a desired altitude or pitch angle, a velocity, and a heading or roll angle. When operating in semi-autonomous mode, the autopilot seeks to meet the desired flight characteristics of the user, who is able to modify these parameters in real-time via radio modem. Navigation decisions are made either by directly observing the airplane or through its on-board video camera. Although a semi-autonomous interface requires a higher user workload than a fully-autonomous waypoint-based interface, many mini-UAV applications require the operator to make strategic decisions on-the-fly, negating the effectiveness of predetermined flight plans and their associated autonomy modes.

An interface that requires the user to directly type in flight parameters such as altitude or roll, as shown in Figure 1, is sufficient for some applications, but leaves much to be desired in many others. In many applications, such precision is neither required nor helpful; rather, it tends to make the UAV more difficult to control. Such interfaces require the user to have a solid understanding of appropriate values for each parameter, and as a result, using this style of interface requires a significant amount of experience and a high amount of user effort. Moreover, the increased precision may come at a cost of lower responsiveness due to operator latency.

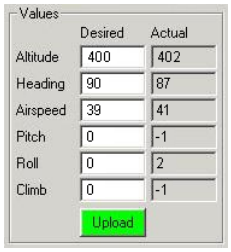

Fig. 1. A numerical-parameter based UAV interface

Numerical parameter-based interfaces are implemented by many research and development teams, as they allow for very precise control and detailed evaluation of the performance of the aircraft. However, such interfaces are very difficult for non-expert users to operate.

Our efforts present an alternative approach, one which trades precision for intuition and speed of response. The goal is an in-the-loop feel that supports efficient task completion. Although no interface can attempt to control the UAV more precisely than directly typing the desired flight characteristics, we claim that such precise control is often unnecessary for a mini-UAV. The user is often content with a lower level of precision, so long as the interface is simple, intuitive, and responsive. This is particularly true for the many applications of these aircraft that involve exploration and search, where a precise *a priori* flight plan is unavailable or unnecessary.

We have created several alternative UAV control schemes in which users operate physical controllers or graphical direct-manipulation interfaces to generate the requisite numerical commands. These interfaces include a direct-manipulation interface for both PDAs and full-size computers, a voice-recognition system, a force-feedback attitude joystick, a force-sensing interface using an IBM TrackPoint™, and a novel "physical icon" interaction

Fig. 2. An in-house fixed-wing semi-autonomous mini-UAV

scheme. A brief overview of our in-house autopilot and UAV platform is given in the following section.

II. UAV and Autopilot technologies

The fixed-wing UAV shown in Figure 2 is constructed of EPP foam and is extremely durable, having survived multiple crashes. The airframe is an in-house design patterned after ZAGI gliders, which are popular dogfight planes in the RC hobby community [4]. The UAVs are powered by an electric motor in a push propeller configuration and are hand launched and belly landed. The plane is actuated by two elevons, with fixed wing tips providing vertical stabilization.

In addition to our in-house designs, we have successfully fitted our autopilot system to a prototype mini-UAV from the Air Force Research Laboratory. This UAV is significantly smaller and lighter than our airframes, and its aerodynamic stability is further decreased due to its very flexible wings. A design requirement of this airplane was the capability of the wings being "rolled up" around the fuselage for easy transportation in small tubular containers. The difficulty of placing control surfaces on "rollable" wings forced a V-tail design with an elevon on each tail. The flight characteristics of such a UAV are dramatically different than our "flying-wing" design, and creating a unified interface for these two widely divergent UAV airframes was a significant focus of our research.

The on-board autopilot is implemented on a single circuit board with a 29 MHz Rabbit microcontroller daughter board, using an Extended Kalman Filter for attitude estimation. On-board sensors include three-axis rate gyros, three axis accelerometers, absolute and differential pressure sensors for altitude and airspeed measurements, and a standard GPS receiver. The autopilot package weighs 2.25 ounces including the GPS antenna, and the size of the autopilot is roughly 3.5 inches by 2 inches. Communication between the airplane and the ground station is accomplished via a low-cost 900 MHz wireless modem.

III. Genearl Interface Considerations

Human-UAV interfaces must seriously consider several factors that tend to be not as critical in ground-based human-robot interfaces:

- The unstable dynamics of a mini-UAV require the interface to support a significant level of autonomy for the UAV to be accessible to many users.
- Many users have little to no experience flying airplanes, and can be confused and disoriented by their many degrees of freedom.
- If the user loses control of the UAV, it may quickly result in significant damage or destruction of the UAV.
- Since the UAV can fly considerable distances away from its operator, depending on the accessibility and hostility of the environment, the UAV may not be recoverable in the event of a crash.

To deal with these challenges and to facilitate simple and intuitive operation of the UAV, we built upon several effective design principles demonstrated by previous work in human-robot interfaces [5]. The interfaces are designed to clearly present the state of the UAV, produce timely feedback, and provide a straightforward mapping between interface controls and the resultant actions of the UAV.

IV. PDA Interface Design and Analysis

Although human-robot interfaces can be implemented on any computing platform, several advantages are offered by a handheld PDA implementation [6]. In particular, the small size and light weight of a PDA make it an ideal platform for the highly mobile environments often associated with mini-UAVs.

In our implementation, the PDA initiates a wireless 802.11b connection to a full-size laptop (the "Base Station") which is connected to a radio modem capable of reaching the airplane within a range of several kilometers. The PDA operator is free to wander wherever he or she wishes while controlling the UAV, as long as the PDA stays within the range of 802.11b communication.

Alternatively, for applications where it is undesirable to create a physical location for the "base station," the required equipment (a full-size laptop, 900 MHz radio transceiver, 2.4GHz video receiver, and respective power sources) can easily be stuffed in a small backpack, with only the PDA in front of the user to coordinate the activities of the low-level equipment. Video monitoring can occur on a eyeglass display. A clean and simple PDA interface, then, offers much more than the initial "gee-whiz" reaction – it completely hides the complexity of the underlying system while greatly increasing its usability.

Although we have not yet measured the relative levels of situational awareness (SA) provided by our UAV interface designs, the key features of the direct-manipulation PDA-UAV interface can be described in terms of the three-tiered definition of SA created by Endsley [7]: perception, comprehension, and projection. The following sections provide a brief analysis of the PDA interface using this framework.

A. Level 1 SA: Perception

The first task of the interface is to help the user *perceive* the current relationship between the UAV and the world, while not overwhelming the user with unnecessary information. We have found that presenting a simplified "wing view" of the UAV successfully abstracts two key characteristics of flight: roll and altitude. This display is shown in Figure 3.

The notion of creating an aeronautical instrument using a miniature airplane icon dates back to the early days

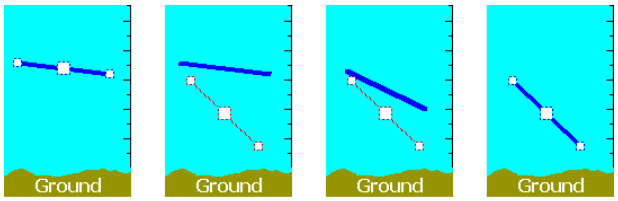

Fig. 3. Direct manipulation of the "wing-view" display. In the upper-left frame, the UAV is holding a relatively high altitude and a slight roll to the right. The upper-right frame shows the user-commanded target attitude. In the lower-left frame, the UAV is progressing to the target, and in the lower-right frame it has reached the new attitude, causing the control handles to return to the telemetry visualization.

Fig. 4. Complete PDA Screen

of aircraft instrumentation [8]. The classic "primary flight display" found in virtually all cockpits, however, has a moving horizon. This makes sense when the pilot is inside the aircraft and experiencing the maneuver first-hand. In contrast, the UAV operator is standing on the ground, and a display with a moving horizon may confuse users not trained as pilots.

The "wing-view" display is created from the vantage point of an observer behind the UAV who is looking through an abstracted cross-section of the UAV's main wing in the direction the UAV is flying. This kind of representation is simple to draw, which is a necessary condition when dealing with the limited graphics hardware of a typical PDA.

Despite its simplistic appearance, the "wing-view" display presents a persuasive abstract visualization of the instantaneous relationship between the UAV and the world. The exact altitude is intentionally not shown; the requirements of the application and the capabilities of the UAV are used to specify a "high" and a "low" altitude for the given application, and the user need not be concerned with the exact altitude target.

Heading and velocity are displayed as gauges already familiar to most users – a compass and a speedometer. As with altitude, a precise scale for the velocity gauge is unnecessary and is intentionally not supplied. This allows the interface to achieve platform independence across a variety of UAV airframes, each of which will have a different range of acceptable velocities. Using an absolute velocity scale would require users to learn more than necessary about the performance characteristics and stall speed of each particular UAV. By using a speedometer only labeled with "min" and "max", the user can be certain that "medium fast" on one UAV also corresponds to "medium fast" on another UAV, even if the absolute velocities in question are quite different. This has proven useful since the same interface has been used to fly both in-house and Air Force UAVs, which have quite different velocity ranges.

B. Level 2 SA: Comprehension

The next level of situational awareness is obtained by combining perceptual (Level 1) data to *comprehend* how these data relate to the overall goal. For a mini-UAV performing a surveillance task, the overall goal is to provide a useful video image of an object of interest. A critical simultaneous task, however, is that the user must keep the UAV in the air. With this in mind, we drew some small mountains and the word "Ground" on the bottom of the "wing-view" display. Furthermore, the "min-max" constraints of the airspeed gauge also help keep the UAV airborne, as it is altogether impossible for the PDA operator to stall the UAV.

Because there are only three instrument displays on the PDA screen, synthesizing their information is a simple matter and does not require a high mental workload. Figure 4 shows the overall layout of the interface, which is intended to present the user with a clean and intuitive panel of data visualizers and controls.

C. Level 3 SA: Projection

The highest level of Situational Awareness described by Endsley is *projection*: the ability to predict what will happen to the system in the near future. By using direct manipulation and visual overlays, the behavior of the UAV is easily predictable. The interface simply maintains constant flight parameters unless it receives a new command, in which case it will seek to match the new parameters. The process of parameter-matching is animated with the visual superposition of "desired" and "actual" state visualizations. On all three displays, "actual" parameters are plotted in blue, and "desired" parameters are plotted in red. The future behavior of the UAV is thus immediately discernible to the user: if any red is showing on the display, the UAV is seeking to match the red visualization. If all displays only have blue visualizations, then the UAV is simply maintaining its current flight parameters.

It has been shown in many complex environments that direct manipulation can be very effective in allowing the user to subjectively generate desired system parameters [9]. Applied to mini-UAV control, this form of interaction can be much more natural than supplying numerical values.

The wing-view display draws three "control handles" on the abstract representation of the UAV, signified by dotted white boxes. The handle in the center of the wing gives the user control of the UAV's target altitude. Using the PDA stylus, the user simply drags the center of the UAV to the desired target altitude. The handle on each wingtip

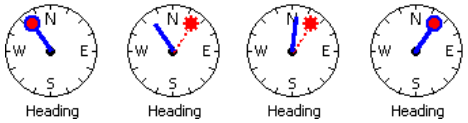

Fig. 5. Direct manipulation of the heading control.

allows the user to drag a wingtip to create a desired roll angle. The interface will not allow the user to create a roll angles beyond the constraints set by the designer – a necessary condition, because most mini-UAV autopilots cannot stabilize an aircraft flying at an extreme roll angle.

Due to communications latencies and the aerodynamics of the UAV, it will take the UAV a minimum of a half-second to meet the user's commands. As previously discussed and shown in Figure 3, target parameters are plotted alongside the current telemetry if a significant difference exists. This approach allows for the user to receive instantaneous feedback from the display and confirmation that the commands have been received. The target airplane is plotted as the stylus moves, without waiting for the real UAV to receive the new target, which further improves the interface response time. When the real UAV's flight characteristics approaches those of the target UAV, the target UAV disappears and the control handles return to the real UAV.

When flying the UAV, the user is directly manipulating the image of the UAV to express a desired relationship between the UAV and world which is quickly matched by the actual UAV. The interface, therefore, moves toward Rutkowski's concept of "transparency" – where the intent of the user is conveyed directly to the world so effortlessly that the interface is no longer a significant factor in the mind of the user [10]. Other researchers have termed this type of interaction "direct engagement," and it results when users have "the qualitative feeling that [they] are directly engaged with control of the objects – not with the programs, not with the computer, but with the semantic objects of [their] goals and intentions." [11]

The heading and velocity gauges have the same functionality and color scheme as the "wing-view" display: the user can drag the velocity or heading "indicator needle" to the desired position, and the airplane will then seek to match the user input.

Due to the difficulties of controlling mini-UAVs that have only two control surfaces, our autopilot cannot simultaneously seek a target roll angle and a target heading; therefore, it is necessary for the interface to determine whether to use the autopilot's "heading hold" mode, or the "roll hold" mode. In our implementation, the most recently issued user command is taken to imply the desired control mode of the airplane. For example, if the user had previously supplied a target roll angle, but then drags the compass needle to signify a target heading, the interface implicitly switches control modes and commands the autopilot to switch to heading-hold mode. The target roll angle is leveled out on the wing-view display to inform the user of this control mode change. This automatic switching of control modes has proven very effective in the real world, and it isolates the user from the underlying complexity of the system.

D. Laptop Implementation

We have ported our PDA interface to run on the standard Win32 platform, using a standard laptop trackpad as the input device. Although a full-size laptop offers significantly higher network and graphics performance, novice users actually found the laptop implementation to be more difficult to use. We suspect that this is due to the difficulty many users have in moving the cursor with a trackpad while holding down a trackpad button to drag a control, as is required by virtually all tasks in our direct-manipulation interface. This activity is quite simple to perform with a PDA stylus, as the pressure of the stylus moving on the display is transmitted to the PDA operating system as the dragging activity that is somewhat difficult to perform on a laptop trackpad. We suspect that an interface that uses single- or double-clicks to select new parameter values, rather than continuous dragging, would be better suited to a laptop trackpad. This observation confirms the fact that interface design must be tailored to its supporting hardware, particularly in time-critical teleoperation systems.

V. Voice Controller

We have implemented a voice controller that can recognize commands such as "Turn Left", "Climb", "Speed up", "Go North", and the like, using a grammar of twenty commands. The speech recognition agent listens for these simple one to three word commands, and when a successful recognition is made, a speech synthesis agent offers instantaneous feedback by simply stating the command in the present progressive tense: "Turning left," "Climbing," "Speeding up," or "Going North." A command packet is then created and sent to the base station server via an open 802.11b connection, and subsequently is translated into a condensed binary packet format and uploaded to the UAV. The process is fast and stable enough that the UAV can be reliably flown using these simple voice commands, either through direct line-of-sight visual contact or by using the UAV's onboard video camera.

VI. Physical Controllers

A. Attitude Joystick Controller

Most joysticks in aeronautical control systems are rate joysticks – that is, a deflection from the resting position of the joystick produces a control surface deflection which under standard conditions will induce a proportional rate of change in the attitude of the aircraft. This control method derives from early aircraft that used mechanical cables to connect the joystick to the ailerons and elevators of the aircraft. In a distributed semi-autonomous control architecture such as presented in this paper, there is no such physical connection, and thus we are free to pursue other options.

We have developed an attitude joystick controller that maps the deflection of the joystick to a deflection in the

aircraft attitude from level flight. Our current parameters for this interface map the x-axis of the joystick to a fixed roll attitude ranging from -45 to +45 degrees, and y-axis deflections map to a fixed pitch attitude from -30 to +30 degrees. This style of interaction is particularly useful to users who are not trained as pilots, as it is completely impossible to cause the mini-UAV to "barrel-roll" from over-deflecting the joystick in the x-axis, and inverting the aircraft by over-deflecting the y-axis is also impossible. In addition, the joystick throttle is mapped to the "min-max" airspeed range discussed previously to eliminate the potential for stalls.

We used a force-feedback joystick to create a haptic sense of situational awareness by causing the force-feedback system to center the joystick in the current attitude of the UAV. Sharp joystick deflections must push against this centering force, but once the UAV meets the newly commanded attitude, the absence of the resistive force informs the user that the UAV has responded to the command, moving the center of the force-feedback space to the new position of the joystick. In practice, we found it useful to slightly bias the force-feedback centering position of the joystick towards the center of the joystick's physical range of motion. This can be easily accomplished by treating the joystick center as the origin of a cartesian plane, and scaling the joystick centering position by a factor slightly less than one. Without this slight bias, a light-handed user may cause oscillations in the system if the force-feedback motors overpower the resistance and inertia of the operator's hand. Furthermore, a slight bias toward the origin of the joystick's coordinate system subconsciously aids users in finding the level flight attitude of the UAV.

B. *TrackPoint™ Controller*

The Twiddler2™ controller[12], shown in Figure 6, is designed for single-hand operation, and features a built-in IBM TrackPoint device positioned under the thumb. This pointing device, found in many laptops, maps 2-dimensional forces to x- and y-axis mouse velocities. Our interface captures the mouse cursor and measures the mouse velocity in each axis, thus approximating the force placed on the TrackPoint by the user's thumb. X-axis forces are mapped to the roll attitude of the UAV and y-axis forces are mapped to the pitch attitude of the UAV in the same fashion as the attitude joystick controller discussed in the previous section. Time averaging the force estimates helps compensate for mouse acceleration and other unwanted operating system effects, and multiplying the deflections of each axis at every timestep by a decay scalar slightly less than one provides a smooth return to the origin when the user stops applying pressure.

Because event-driven operating systems simply stop sending messages when movement of the mouse cursor stops, the decay scalar allows the interface to handle the intermittent flow of data, treating the lack of data as an indication that forces are no longer being applied to the TrackPoint device. After experimentation, we settled on a time-averaging constant that allows the user to achieve full deflection on the interface with approximately one-half second of strong pressure on the TrackPoint. The decay scalar was set so that the interface returned to zero deflection approximately one second after the user completely removed pressure from the TrackPoint device. These constants help smooth the command stream being sent to the UAV while still capturing the intent of the user.

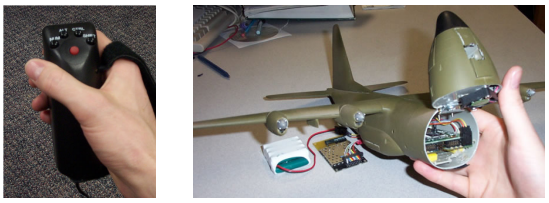

Fig. 6. Single-handed operation of the Twiddler2 Controller (left), and the physical icon opened to show the placement of its onboard autopilot (right). The other components on the table are its radio modem and power source.

C. *Physical Icon Interface and Mixed Reality*

We created a novel Human-UAV interaction scheme by using a toy airplane as a "physical icon" of the real UAV. Inserting an autopilot, radio modem, and power source into the physical icon allows it to continuously track its own attitude in three-dimensional space. These measurements are sent via a small 900MHz radio modem to a nearby computer, which converts the attitude of the physical icon into roll attitude and pitch attitude commands for uploading to the real UAV. The UAV is surprisingly easy to control using this interaction scheme, as the user is able to maintain high situational awareness because he or she is literally holding a physical representation of the UAV. In steady-state conditions, the physical icon serves as a model of the UAV state, assisting the user in the comprehension (Level 2) phase of situational awareness discussed previously. Immediately after the user changes the attitude of the physical icon, it becomes part of the user's projection of the near-future behavior of the UAV, corresponding to the third level of situational awareness.

To assist the user in differentiating between these two roles of the physical icon, we have created a graphical interface similar to the PDA interface discussed previously. Telemetry from the UAV is used to orient a blue-colored OpenGL 3D airplane model. The attitude of the physical icon is used to orient an identical red-colored model. After a command has been received and matched by the UAV, the telemetry of the UAV matches the orientation of the physical icon and the two models merge to one. Immediately after the user alters the attitude of the physical icon, the red model serves as the target while the blue model tracks the progress of the real UAV in matching the desired orientation.

To further increase the functionality of this interface, we introduced an element of mixed reality [13] by superimposing the two OpenGL airplane models onto a horizon-stabilized video feed produced by digitizing and rotating

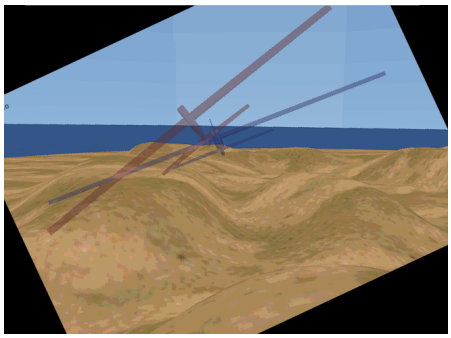

Fig. 7. Mixed-Reality Physical Icon interface. The actual telemetry, plotted as a transparent blue OpenGL model, is shown slightly rolling to the left. The user has requested a climb and a sharper left roll, as shown in the second OpenGL model, which is transparent red when seen in color. The simulated video image has been rolled so as to level the horizon.

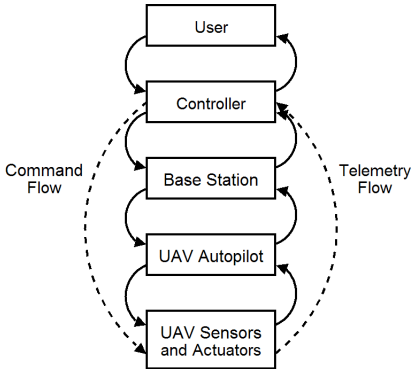

Fig. 8. Command and Telemetry Data Flow

the UAV's analog video feed by the inverse roll angle of the airframe (Figure 7). Because the camera is rigidly mounted to the airframe, this simple transformation helps stabilize the video feed and allows the virtual-reality models to appear as if they belong in the camera frame itself. The user has the impression that the UAV's onboard camera is instead a "chase camera" that has a constant up-vector, regardless of the bank angle of the real UAV. Because the user need not attempt to perform the horizon-stabilizing rotation though mental effort, the user is able to focus more energy on analyzing the content and information in the video stream.

VII. System Architecture

We have created a modular architecture which can support a variety of controllers. The flow of sensor data and user commands utilizes a hierarchal architecture similar to that developed in previous work [14], and is illustrated in the following diagram. Actual data flow is illustrated by solid lines, and the user's mental model is illustrated by the dotted lines.

The influence of the system image and the subsequent formation of the user's mental model has been demonstrated in previous work [15]. The high-level controllers give users the impression that their commands are directly actuating the UAV, the overall goal of the system is to support this oversimplified mental model of the command and sensor data flow.

The base station, running on a full-size laptop, starts a server which accepts wireless 802.11b connections from external controllers, which may be physically or virtually separated from the base station process. These external controllers communicate with the server running on the base station to gain indirect access to the radio modem of the base station, thereby allowing the external controllers to indirectly communicate with the UAV. The base station server accepts and transmits a simple set of commands.

The base station server adds a required element of stability. If a client controller were to stop working, the base station can still control the UAV using a numerical parameter-based interface. In addition, the base station ensures that the user is giving reasonable commands, and can "veto" unreasonable requests, such as negative altitudes, negative velocities, and the like.

VIII. Interface Evaluations and Observations

Physical UAVs have been flown extensively with each interface to gather anecdotal evidence, and simulations were used to carefully quantify and reinforce the anecdotal evidence. Testing in simulation has utilized both software autopilots and a mixed hardware-software scheme in which telemetry from a simulator is transmitted through an RS-232 serial connection to the hardware autopilot, which ignores its own (stationary) sensor readings and instead reacts to the telemetry streaming from the simulator. Control surface deflections produced by the autopilot are then transmitted up the RS-232 interface and used by the virtual UAV in the simulator. The open-source simulator we used implemented the second- as well as fourth-order Runge-Kutta methods[16].

As previously discussed, our UAV fleet includes both flying-wing and V-tail UAVs. From the perspective of the high-level interface, switching between these airframes is completely transparent. These UAVs range in wingspan from 2 to 5 feet, with the largest UAV weighing approximately 5 pounds and the smallest weighing well under one pound. By placing all airframe-specific calculations on the base station and the on-board autopilot, the only difference the high-level user notices when switching between airframes is that larger and heavier UAVs take more time to change altitude and velocity. As a result, operating the UAV using a high-level semi-autonomous interface is simple enough that people who have had absolutely no piloting experience are quickly able to learn the interface and fly the airplane wherever they wish.

To provide a quantative comparison of the relative difficulty of issuing commands with our interfaces, we created a testing scheme in which users flew virtual UAVs in a simulator which was capable of generating a real-time perspective identical to that produced by a mini-UAV's onboard camera. The simulator was programmed to superimpose high-level commands in large letters on the simulated camera feed, such as "Turn Left!" or "Climb!" The simulator implemented safeguarding by comparing the near-future location of the UAV with the terrain so as to

avoid requesting commands that would lead to a crash. Log files were generated that recorded the time required for the user to make each requested change in the UAV state.

Users were thus required to focus their attention on the simulated camera image, extract a high-level command from the camera view, interact with the high-level interface, and return focus to the simulated camera view for confirmation of the success of the command. This interaction cycle is nearly identical to that performed by users performing a tracking or searching task with a real UAV. Median user response times function as estimates of the relative difficulty of issuing a command via a particular interface. Four novice users with no piloting experience participated in this study, and the relative response times were virtually identical for all users. The following table summarizes these results.

TABLE I
MEDIAN TIME REQUIRED FOR ISSUANCE OF COMMANDS

Interface	**Median Time (sec)**
Attitude Joystick	1.44
Physical Icon	1.61
Attitude TrackPoint™	1.69
Voice Recognition	2.39
PDA-Based Direct-Manipulation	2.62
Laptop-Based Direct-Manipulation	2.73
Numerical Parameter Entry	7.52

The interfaces can be grouped into four categories which describe both their functionality and account for their median response times. We summarize our user tests and offer our impressions of the real-world feasibility of each interface.

A. Physical Interfaces

The first three interfaces in the table have the fastest response times, and they all map a physical motion of the user to a desired attitude of the UAV. This mapping is simple, straightforward, and easily understood by the user. In addition, these interfaces do not require a focal shift between the onboard camera feed of the UAV and the interface itself.

However, interaction time is certainly not the only factor by which to judge the efficacy of an interface. The very fact that these interfaces require physical components could be detrimental or even prohibitive, depending on the application. A UAV control station built into a vehicle such as a Humvee would have ample space and power for any of the physical interfaces, and an attitude joystick or physical icon interface would offer fast and easy control of the UAV. However, in a wilderness search-and-rescue application, the UAV interface might need to be transported on a very small vehicle such as an ATV. In this context, the physical requirements of this class of interface would be a limiting factor, and a PDA-based interface would save size and weight.

B. Voice Recognition

In situations where the UAV is flown solely using visual contact, the direct-manipulation interfaces do not seem to adequately address the problem of tracking the airplane, as the focal shift from standard reading distance to the several hundred yard (or more) distance to the UAV is very difficult to perform reliably. Physical interfaces are also problematic, as the control mappings must be inverted when the UAV is flying directly at the user. We found the voice controller to be much more effective in this application, as users can completely devote their visual attention to tracking the airplane and allow the headset microphone to pick up and transmit commands. Commands such as “Go South” are environment-relative, as opposed to the UAV-relative commands produced by physical controllers, and thus need not be inverted when the UAV flies toward the operator.

Real-world tests with this interface have demonstrated that ambient wind noise and conversation can wreak havoc on the reliability of the voice-recognition system. A method of “muting” the microphone input is required, but even with such a system in place, considerable difficulties arise in environments with strong winds, and a moving vehicle would present similar difficulties. However, our experience has shown the voice interface to be very valuable, under favorable weather conditions, when the UAV is flown by line-of-sight navigation.

C. Direct Manipulation

A direct-manipulation interface offers a compromise between the response times of the physical interfaces and the numeric parameter-based interface. Perhaps its greatest strength is its neglect tolerance – the ability of the interaction scheme to function as user attention decreases[17]. A physical interface such as a physical icon or attitude joystick requires the user to literally hold the desired attitude command. The direct manipulation scheme, by contrast, allows the user to symbolically express the desired attitude. No further interaction with the interface is necessary until a change in attitude is desired. As a result, this type of interaction scheme is much better suited to applications where the user has other pressing concerns, such as navigating an off-road vehicle. In addition, this scheme is better suited to multi-agent teleoperation, as several direct-manipulation panels could simultaneously display state and accept commands for several UAVs.

D. Numerical Parameters

Of course, users can fly UAVs via traditional parameter-based interfaces, but they have several important drawbacks that are addressed by the high-level interfaces previously discussed. First and most obviously, parameter-based interfaces require users to type. In cold- or adverse-weather conditions, typing may be very difficult, if not impossible. Typing also requires the keyboard or keypad to be supported by either a stationary surface or a user’s free hand, and requires at least one hand for parameter entry. In highly mobile environments, such stability may be infeasible.

Second, users of parameter-based interfaces must perform additional steps of mental arithmetic, such as incrementing or decrementing parameter values. Informal tests with many users have demonstrated that many people have difficultly performing high-stress, time-critical arithmetic problems, no matter how trivial they may seem. This observation is confirmed by the fact that our formal test users required three to five times as much time to enter numerical parameters than to express the same commands on a direct-manipulation interface or a physical controller.

In short, traditional numerical interfaces to semi-autonomous UAV control modes leave much to be desired. Although they are excellent for high-accuracy testing and evaluation of UAV hardware and software, they are not ideal for end-users with little to no flying experience.

E. Summary

The observations of the preceeding sections can be summarized by three factors that help determine the effectiveness and applicability of a human-UAV interface: precision, mobility, and responsiveness to user input. In the following table, we use the same categorization of interfaces used in the preceeding sections: the joystick, physical icon, and TrackPoint interfaces are abbreviated as simply "Joystick," the direct-manipulation interface is abbreviated as "PDA," and the voice-recognition and numerical parameter interfaces are abbreviated as "Speech" and "Numeric," respectively.

	Precision	**Mobility**	**Responsiveness**
High	Numeric	PDA	Joystick
	Joystick	Speech	PDA
Low	PDA	Numeric	Speech
	Speech	Joystick	Numeric

IX. Future Work

Our development efforts and field studies suggest several areas of future work. We intend to extend our work in video stabilization and processing by incorporating a live video image from the camera into the direct-manipulation interfaces. This will require some sort of a mixed-mode display and perhaps the ability to switch between a full-screen camera view and a separate window that will contain a small version of the camera feed and all the controls necessary to fly the UAV. Waypoint-driven interfaces could also be incorporated into this mixed-mode display. We would also like to experiment with superimposing a transparent image of a compass on the image of a downward-facing UAV camera, and allowing heading commands to be generated simply by tapping or circling an object of interest with a stylus.

Many of our field tests occurred on the edge of a mountain valley, and have demonstrated a potential need for higher automation of altitude tracking. Specifying a "medium-high" altitude on a high-level semi-autonomous interface equates to an altitude of approximately 500 feet above the launch point – but if nearby mountains rise several thousand feet above the launch point, flying directly at these mountains at a launch-relative "medium-high" altitude would be disastrous. In contrast, using *a priori* topographic maps, a terrain model, and GPS localization, it would be possible to have the UAV automatically fly at a ground-relative altitude.

X. Conclusions

Although parameter-based semi-autonomous UAV teleoperation interfaces offer great precision, they do so at the expense of high user workload and a steep learning curve. Searching of some form is required for many applications of mini-UAVs, and predetermined waypoint-based flight plans are difficult to prepare for such missions. Moreover, the flight plan may have to be heavily modified or forgotten altogether as the mission progresses. In situations such as these, neither parameter-based nor waypoint-based interfaces seem to offer the ideal control methodology. We have presented and evaluated several interaction schemes which serve to isolate the user from the technicalities and arithmetic requirements of the underlying parameter-based control system. We conclude that a high-level interaction scheme drastically simplifies the typical commands required for searching tasks using a mini-UAV, especially when the interface design is tailored to the physical and cognitive requirements of the application.

References

[1] T. Coffey and J. Montgomery, "The emergence of mini UAVs for military applications," in *Defense Horizons*, no. 22, December 2002.

[2] O. of the Secretary of Defense, *UAV Roadmap*, 2002, pp. 127–133.

[3] D. Kingston, R. Beard, T. McClain, M. Larsen, and W. Ren, "Autonomous vehicle technologies for small fixed-wing UAVs," in *AIAA 2nd Unmanned Unlimited Systems, Technologies, and Operations – Aerospace, Land, and Sea Conference and Workship & Exhibit, San Diego, CA*, September 2003, paper no. AIAA-2003-6559.

[4] Http://zagi.com.

[5] T. W. Fong, C. Thorpe, and C. Baur, "Advanced interfaces for vehicle teleoperation: Collaborative control, sensor fusion displays, and web-based tools," in *IEEE International Conference on Robotics and Automation*. IEEE, April 2000.

[6] T. W. Fong, C. Thorpe, and B. Glass, "Pdadriver: A handheld system for remote driving," in *IEEE International Conference on Advanced Robotics 2003*. IEEE, July 2003.

[7] M. R. Endsley, "Toward a theory of situation awareness in dynamic systems," in *Human Factors*, vol. 37, March 1995.

[8] O. E. Patton, *Aircraft Instruments: Their Theory, Function and Use*. D. Van Nostrand Company, 1941, pp. 110–114.

[9] B. Shneiderman, *Designing the User Interface, Third Edition*. Addison-Wesley, 1998, pp. 186–233.

[10] C. Rutkowski, "An introduction to the human applications standard computer interface," in *BYTE*, October 1982, pp. 299–300, cited in Schneiderman, p. 202.

[11] E. Hutchins, J. Hollan, and D. Norman, "Direct manipulation interfaces," in *User-Centered System Design*, D. Norman and S. Draper, Eds. Lawrence Erlbaum Associates, 1986.

[12] Http://www.handykey.com.

[13] R. T. Azuma, "A survey of augmented reality," in *Presence: Teleoperators and Virtual Environments*, vol. 6, no. 4, August 1997.

[14] N. R. Council, *Uninhabited Air Vehicles: Enabling Science for Military Systems*. National Academy Press, 2000, ch. 7.

[15] D. A. Norman, *The Design of Everyday Things*. Currency and Doubleday, 1988, pp. 17, 189–194.

[16] The open-source Slope Soaring Simulator, written by Danny Chapman, is available at http://www.rowlhouse.co.uk/sss/.

[17] J. W. Crandall, C. Nielsen, and M. Goodrich, "Towards predicting robot team performance," in *2003 IEEE Conference on Systems, Man, and Cybernetics (SMC '03)*.