I suggest that we incorporate Spencer Clark's work. This means taking text from his thesis and adding it here, presenting it in the broader context of your work. Since he explicitly incorporates order constraints and no-fly zones, it is important validation for your framework. I don't think you'll need to describe his algorithms any more than you need to describe your algorithms, since the algorithms were described in a published conference paper.

Sliding Autonomy for UAV Path Planning: Adding New Dimensions to Autonomy Management

Lanny Lin, *Member, IEEE,* and Michael A. Goodrich, *Senior Member, IEEE*
Computer Science Department
Brigham Young University
lanny.lin@byu.edu, mike@cs.byu.edu

***Abstract*—Increased use of autonomy also increases human-autonomy interaction and the need for humans to manage autonomy. We propose a new autonomy management approach, sliding autonomy, where the user can influence the behavior of the autonomous system along three new dimensions: information representation, task constraints (spatial), and time allocation (temporal). We analyze how this approach fits in the integration challenges guideline we identified in our prior work and apply it to the task of UAV (Unmanned Aerial Vehicle) path planning to support Wilderness Search and Rescue (WiSAR). We evaluate the usefulness of the approach against manual and simple pattern path planning methods with a user study. Results show that the sliding autonomy approach performs significantly better than the other two methods without increasing the users' mental workload, and the performance of the human-autonomy team outperforms either human or autonomy working alone. We also discuss some interesting observations from the user study.**

***Index Terms*—Unmanned aerial vehicles, path planning, navigation, adjustable autonomy, supervisory control**

I. Introduction

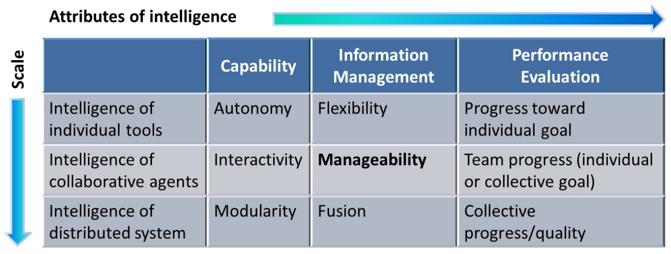
WITH the rapid advancement in technology, people are seeing increased use of autonomy to augment human abilities and support human decision-making in many application domains (e.g., [1–4]). At the same time, increased use of autonomy also means increased human-autonomy interaction and increased need for human to manage autonomy [5]. Even for fully autonomous systems, human input can potentially improve the system's performance and safety. The managerial responsibilities include monitoring the safety of the autonomous system, supervising autonomy to achieve acceptable performance, and making sure autonomy is working toward the collective goal of the overall system. The human operators are domain experts who can use domain-specific knowledge to assist the autonomous system when it deals with changing environments, uncertainty, and case-specific scenarios. However, the humans in such interactions are not likely the designers of the autonomous systems, but these humans must still manage autonomy, because "only people are held responsible for consequences (that is, only people can act as problem holders) and only people decide on how authority is delegated to automata" [6, 7]. Therefore, it is necessary to design tools and interfaces that enable human users to manage the autonomous behaviors of the system efficiently and effectively; such tools can improve task performance and the experience of the human operator in human-autonomy interaction.

We propose a new autonomy management approach where the user can influence the behavior of the autonomous system along three new dimensions: 1) **Information Representation**: user can modify information representation relate to the problem that autonomy can understand, such as areas of focus and task-difficulty; 2) **Task Constraints**: user can choose where to add constraints to the problem to change its properties; and 3) **Time Allocation**: user can decide how much time to allocate to a subtask out of the total task time. Once the information representation and task constraints are set, the user can move a slider to control how much time is allocated to the subtask and see immediately how the action affects the solution generated by autonomy. This is why we named our approach **Sliding Autonomy**. Full solution to the problem is combined from solutions of all the subtasks. It is most likely different from a solution generated by full autonomy because of the additional human-autonomy interaction. Ideally, humans should be doing what humans are good at, and autonomy should be doing what autonomy is good at (in reality, autonomy is also doing what humans do not want to do). The human-autonomy team should perform better than human or autonomy working alone.

Many approaches to autonomy management already exist, such as *Supervisory Control* [8], *Mixed-initiative* [9], *Collaborative Control* [10], *Adjustable Autonomy* [11, 12] (also referred to as *Sliding Autonomy* [13] or *Adaptive Automation* [14, 15]). The approach we propose falls under the category of *Adjustable Autonomy*. The three dimensions we identified are in addition to dimensions of *Adjustable Autonomy* identified by Bradshaw et al. in [16].

In our previous work [3] we identified key elements of autonomy integration challenges along two dimensions: *attributes of an intelligent system* (capability, information management, performance evaluation) and *scale* (individual versus group), which can serve as a guideline in designing autonomous components and autonomy management tools. In this paper we extend this guideline to include attributes needed when human and autonomy work collaboratively. We apply the proposed approach to the task of UAV (Unmanned Aerial Vehicle) path planning to support Wilderness Search and Rescue (WiSAR), describe what these three new dimensions mean in this context, and explain how they fit in the guideline we defined.

Camera-equipped mini-UAVs can be useful tools in WiSAR operations by providing aerial imagery of a search area with the benefits of quick coverage of large areas, access of hard-


Attributes of intelligence
Scale

	Capability	Information Management	Performance Evaluation
Intelligence of individual tools	Autonomy	Flexibility	Progress toward individual goal
Intelligence of collaborative agents	Interactivity	**Manageability**	Team progress (individual or collective goal)
Intelligence of distributed system	Modularity	Fusion	Collective progress/quality

Fig. 1. Autonomy integration challenges defined along two dimensions. Horizontal dimension: attributes of intelligence. Vertical dimension: scale.

to-reach areas, and lower cost than manned aircraft [17, 18]. In fact Canadian mounties claim that they have successfully saved a person with a police drone in a recent rescue mission[1]. UAV path planning is an important task because a good flight path can increase the probability of finding a missing person by making efficient use of the limited flying time. Various algorithms have been developed to generate UAV paths automatically (e.g., [19–21]). By applying sliding autonomy to this path planning task, we argue that this approach:

- enables the domain expert user to incorporate information only available to or understandable by the user;
- is easy to understand without knowing how autonomy works behind the scene;
- lets the human do what human is good at (planning strategically) and autonomy do what autonomy is good at (planning tactically), resulting in better performance than human or autonomy working alone;
- enables the user to align task goal with system goal;
- and improves human's experience during the human-autonomy interaction.

To evaluate the usefulness of the proposed approach, we performed a user study and compared the sliding autonomy method against two other planning methods (manual and simple pattern path planning) in two WiSAR scenarios (a synthetic scenario and a real scenario). We measured each user's performance with each method and also the user's performance on a secondary task (answer questions in a group chat window). Experiment results show that the sliding autonomy method performed significantly better than the manual or simple pattern planning methods with no increased mental workload. The human-autonomy team also performed better than the human or autonomy working alone.

In section II we explain how the proposed approach fits into the extended autonomy design guideline and describe how a user can manage autonomy along each of the three new dimensions in the context of UAV path planning. Section III covers related work in literature. Section IV lists our hypotheses followed by user study design in section V. Then we present experiment results in section VI and discuss our observations in section VII. In section VIII we conclude the paper and list possible future work.

[1]http://www.theverge.com/2013/5/10/4318770/canada-draganflyer-drone-claims-first-life-saved-search-rescue

II. Autonomy Design Guideline and New Dimensions

A. *Autonomy Design Guideline*

In our previous work [3] we organized the challenges of autonomy and management tool design along two dimensions: *attributes of an intelligent system* (capability, information management, performance evaluation) and *scale* (individual versus group), which can serve as a guideline in designing autonomous components and autonomy management tools. Here we extend this table by adding a row in the middle describing what attributes are needed when multiple agents work collaboratively (see Figure 1). A human-autonomy team working on the same task falls within this category. As an individual tool, an autonomous component needs to be able to perform a task (**Autonomy**); the operator can match capability to task according to the information available to the operator, which requires that autonomy can be interrupted, paused, aborted, and resumed (**Flexibility**); the performance is evaluated to match individual task goal. When a human-autonomy team works on the same task collaboratively, the autonomous component needs to provide interfaces so the human can influence the autonomous behavior with human inputs (**Interactivity**); the human agent should be able to manage how autonomy works in order to jointly find a solution by utilizing information only available to the human agent and/or feed information to autonomy in a representation that autonomy can understand (**Manageability**); and when performance is evaluated, the human operator can judge whether the individual goal aligns with the collective goal of the system. As part of a larger distributed system, each autonomous component needs to be modular (**Modularity**), so they can be mixed and matched to support different user roles; information from various sources need to be combined and presented to one or multiple users in a **Fusion**; and performance of the system needs to be evaluated as a whole. This paper focuses on the middle row of this guideline: intelligence of collaborative agents (human-autonomy team). The three dimensions we propose are ways path planning autonomy can be managed, and the path planner component is designed to accept human inputs along the three dimensions to provide interactivity. The human can also incorporate information from various sources and influence the behavior of path planning autonomy, making sure the task goal aligns with the ultimate goal of finding the missing person quickly.

B. *Information Representation Dimension*

Many path planning algorithms use a probability distribution map that shows where are likely places to find the missing person. We also designed our path planning algorithms to support a task-difficulty map: a spatial representation showing (one minus) sensor detection probability in different parts of the search region. For example, probability near the last known position (LKP) of the missing person is normally high (low task difficulty); and the probability of detecting the missing person in a dense vegetation area from camera footprints is normally low (high task difficulty). The objective of path planning is to find a path that maximize the detection

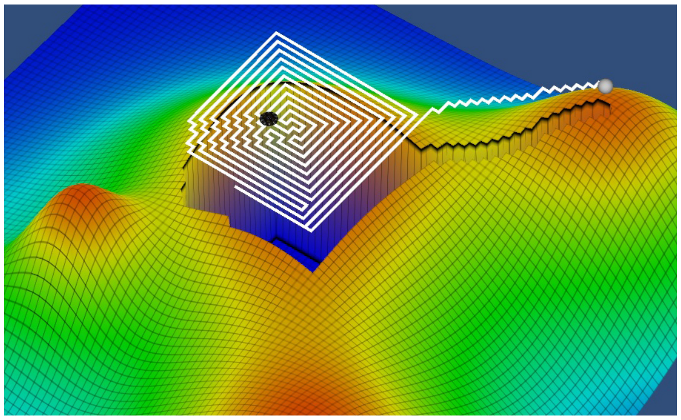


Fig. 2. A screen capture of the sliding autonomy tool showing a 20 minute path. The 3D surface shows the probability distribution map. The UAV icon in the middle indicates the start point of the path segment and the sphere on the right indicate the end point.

probability of the missing person given a fixed flying time. The maps can be systematically generated based on terrain features and vegetation data [21, 22]. However, the searcher likely wants to include his/her domain expertise (past experience, knowledge of the search region, etc.) and additional information (maybe new evidence found during the search) in the planning. These types of information are not directly understandable by autonomous algorithms, but the searcher can incorporate them into the probability distribution map and the task-difficulty map using map editing tools[2], thus influence the behavior of path planning autonomy. Marking an area with high probability, the searcher indirectly tells the UAV to treat the area with high priority; marking an area with high task-difficulty, the UAV might make multiple passes over the area to search more thoroughly. The 3D surface in Figure 2 shows an example probability distribution map where hills (red) indicate high probability and the flat area (blue) indicates low probability.

By managing information representation, the searcher can quickly figure out how his actions will affect the behavior of autonomy, even though he/she has no idea about how autonomy works behind the scene. Our interface supports autonomy management along this dimension. But we choose to not include it in the current user study (due to the one-hour limit per test subject) and leave the evaluation in a separate user study.

C. *Task Constraints Dimension*

The searcher can also influence the behavior of path planning autonomy by adding constraints to the problem. Constraints can be the start/end points of the path or no-fly zones. Setting an end point in an area is a way for the searcher to indirectly tell path planning autonomy that he/she wants the UAV to search this area. For example, if a piece of clothing is found by the ground team, the searcher can force path planning autonomy to go visit that area by setting an end point there. The searcher can also use this method to direct path planning autonomy to not visit an area (maybe the area has been thoroughly searched by the ground team) by setting the end point in other areas. By managing autonomy along this dimension, the searcher can incorporate additional information, information not directly understandable by autonomy, to improve task performance and align the task goal with the overall goal of finding the missing person quickly. A no-fly zone is pretty straight forward way to restrict the UAV from visiting certain areas maybe due to safety reasons. In Figure 2, the UAV icon in the middle indicates the start point of the path segment and the sphere on the right side indicates the desired end point. Because start/end points and no-fly zones are all spatial constraints, the searcher manages autonomy along a spatial dimension.

Spatial constraints are easy to understand, so the searcher knows how these constraints will affect the behavior of path planning autonomy. In our user study we fixed the start point of the path to the center of the map because that was the last know position of the missing person. The searcher can set the end point for the current path segment anywhere on the map, and this end point automatically becomes the start point for the next path segment. No-fly zone was not included in this user study (it was tested in a separate user study [23]).

D. *Time Allocation Dimension*

Once the information representation and task constraints (optional) are set, the searcher can use a slider to allocate different time to path planning autonomy. For example, as shown in Figure 2, for a 60-minute total flight, the searcher can set an end point to the probability hill on the right, and then move the slider to see immediately what path segment autonomy would suggest. The path segment shown is when 20 minutes are allocated. If the searcher is happy with the suggestion, he approves the path segment. The UAV moves to the end point and "vacuums up" the probability along the path (how much can be vacuumed up is determined by the task-difficulty map). Then the searcher works together with autonomy to plan the path for the remaining 40 minutes. The two (or more) path segments are joined to form the final path. In the example shown, the combination of the information representation, task constraints, and time allocation allows the searcher to cover the middle area pretty well and then move to the search area on the right ready to search there. If no constraint is set, autonomy might decide to search the probability hill on the left instead. If more flight time is allocated with the set end point, autonomy might start searching the area on the right. Because the searcher decides how many path segments to plan and how much time to allocate to each path segment, the searcher manages autonomy along a temporal dimension.

As the searcher moves the slider to control time allocation, the sliding autonomy tool provides suggested path segment immediately. This instant feedback provides the searcher the ability to perform "what-if" analysis and see the causal effect between his/her action and changes in autonomous behavior.

By managing time allocation, the searcher can break the path planning task into subtasks. The objective of path planning autonomy is to maximize the amount of probability the

[2]Such as these tools at http://tech.lannyland.com/demos.html.

UAV can collect following the current path segment. The objective of the human or the entire distributed system (where the UAV is only a part of the operation) is to find the missing person quickly. By setting constraints and change the amount of time allocated to a path segment, the searcher has the ability to incorporate additional information into the problem, and align the task goal with the overall system goal. The ability to break the task into subtasks also enables the searcher to plan more strategically (prioritizing areas in the entire search region) while autonomy works more tactically (covering the current search area well). Ideally such a human-autonomy team should work better than either human or autonomy working alone.

III. Related Work

Drucker defines automation as a "concept of the organization of work [24]." Goodrich and Schultz [25] define the HRI problem as "understanding and shaping the interactions between one or more humans and one or more robots." They also specified robot-assisted search and rescue as a key area for HRI research. In their 1978 seminal paper [26], Sheridan and Verplank propose the idea of a *Level of Autonomy* spectrum, with full teleoperation at one end and full autonomy at the other. In the middle, the robot could suggest actions to humans or make decisions before informing humans. Parasuraman et al. [27] extended this one-dimensional spectrum to four different broad functions: information acquisition, analysis, decision selection, and action implementation. In [8] Sheridan proposes *Supervisory Control*, in which a human divides the task into a sequence of subtasks that the robot is capable of performing, and the human then provides guidance when the autonomous system cannot solve a problem on its own. In contrast to the top-down philosophy of supervisory control, a *Mixed-initiative* approach advocates the idea of dynamically shifting tasks when necessary [9]. *Collaborative Control*, which can be thought of as an instance of mixed-initiative interaction, is a robot-centric model; instead of the human always being in-charge, the robot is treated as a peer and can make requests to humans through dialogs [10]. *Adjustable Autonomy* [12] (also referred to as *Sliding Autonomy* [13] or *Adaptive Automation* [14]) is another type of mixed-initiative interaction, one that enables the human-automation team to dynamically and adaptively allocate functions and tasks among team members.

Dorais et al. [11] discuss a framework for human-centered autonomous systems for a manned Mars mission. The system enables users to interact with these systems at an appropriate level of control but minimize the necessity for such interaction. Bradshaw et al. discuss principles and pitfalls of adjustable autonomy and human-centered teamwork, and then present study results on so-called "work practice modeling" and human-agent collaboration in space applications [28]. In [29] Kaber et al. describe an experiment simulating an air traffic control task where manual control was compared to Adaptive Automation (AA). Results suggest that humans perform better with AA applied to sensory and psychomotor information-processing functions than with AA applied to cognitive functions; these results also suggest that AA is superior to completely manual control. Brookshire et al. present preliminary results for applying sliding autonomy to a team of robots performing coordinated assembling work to help the system recover from unexpected errors and to thereby increase system efficiency [30]. Dias et al. identified six key capabilities that are essential for overcoming challenges in enabling sliding autonomy in peer-to-peer human-robot teams [13]. Bradshaw et al. [16] propose two dimensions of Adjustable Autonomy (descriptive and prescriptive) to address the two senses of autonomy (self-sufficiency and self-directedness) and discuss how permissions, obligations, possibilities, and capabilities can be adjusted. Bradshaw et al. [7] also summarized some widespread misconceptions on autonomy and listed seven deadly myths of "autonomous systems."

Human is an integral part of the human-autonomy team. When working with autonomy, the human often takes on the supervisor role. Bainbridge points out that automation requires the human operator to take additional management responsibilities [5], and Sartar identified in [31] two automation management policies: *management by consent* and *management by exception*, defining whether the human always retain authority or can the system take initiative. For complex automations, the human tends to rely on his/her *mental models* (defined by Norman in [32]) to manage the system.

UAV technology has emerged as a promising tool in supporting WiSAR [17, 19]. The goal of our research is to support fielded missions in the spirit of Murphy's work [2]. Many path planning algorithms in the literature address obstacle avoidance while planning a path to reach a destination using A* [33], LRTA* [34], D* [35], Voroni diagrams [36, 37], or probability roadmaps and rapidly-exploring random tree (RRTs) [38]. Hierarchical heuristics approaches were also developed, such as Hierarchical A* (HA*) by Holte et al. [39], hierarchical task-based real-time path planning by Naveed et al. [40], and Hierarchical-AO* (HiAO*) by Meuleau and Brafman [41]. In [42, 43] Bourgault et al. describe how to use a Bayesian model to create paths for a single UAV or multiple coordinated UAVs to maximize the amount of probability accumulated by the UAV sensors. The algorithms we used in this paper are algorithms designed from our previous work [20, 21] using techniques such as global warming technique, convolution, Gaussian mixture models, and Evolutionary Algorithm.

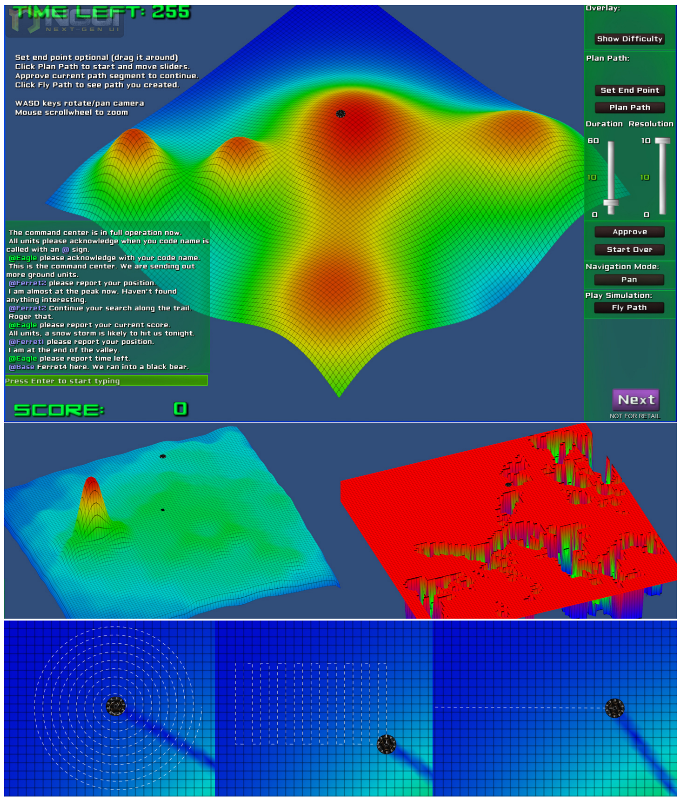
IV. Hypotheses

We performed a user study to evaluate the usefulness with our proposed sliding autonomy approach. More specifically we verify the following hypotheses:

H1: Sliding autonomy method performs better than either the manual path planning method or a semi-autonomous simple pattern path planning method in both low information and high information scenarios.

H2: Sliding autonomy method performs better than autonomy working alone in both low information and high information scenarios.

H3: Sliding autonomy method does not increase the mental workload of the operator when compared against manual and pattern methods.



Set end point optional (drag it around)
Click Plan Path to start and move sliders.
Approve current path segment to continue.
Click Fly Path to see path you created.

WASD keys rotate/pan camera
Mouse scrollwheel to zoom

The command center is in full operation now.
All units please acknowledge when you code name is called with an @ sign.
@Eagle please acknowledge with your code name.
This is the command center. We are sending out more ground units.
@Ferret2 please report your position.
I am almost at the peak now. Haven't found anything interesting.
@Ferret2 Continue your search along the trail.
Roger that.
@Eagle please report your current score.
All units, a snow storm is likely to hit us tonight.
@Ferret1 please report your position.
I am at the end of the valley.
@Eagle please report time left.
@Base Ferret4 here. We ran into a black bear.
Press Enter to start typing

Fig. 3. Top: User study simulation interface with sliding autonomy method showing the probability distribution map for scenario 1. Middle left: Probability distribution map for scenario 2. Middle Right: Task-difficulty map for scenario 2. Bottom: The three patterns available to user in pattern planning mode, spiral, lawnmower, and line.

V. User Study Design

We performed a 2×3 within-subject design with 2 scenarios (easy vs difficult) and 3 planning methods (manual, pattern, and sliding autonomy). All participants completed all 6 exercises. We counterbalanced the order of the scenario and planning methods to reduce learning effect. Half of the participants started with scenario 1 (the other half scenario 2). and the order of the planning methods were randomly drawn without repeat from the permutation of all possible combinations (without following the same order in both scenarios).

A. Participants

After analyzing data collected from a pilot study with 6 volunteers, it was determined that 25 participants would likely produce significant test results. We recruited a total of 26 college students (14 male males and 12 females) between the age of 19 and 30 (average 22.89). None has colorblindness. The majority has no experience with robots (57.69%), and 34.62% of them are slightly experienced with vacuuming robots.

B. Simulation Environment

The user study is conducted in a 3D simulation environment. The top portion of Figure 3 shows a screen capture of the simulation interface. Both the probability distribution map and the task-difficulty map are displayed as 3D surfaces with a color map (red means high altitude and blue means low). The user can switch between the two maps anytime. The user can also rotate/pan a map and zoom in/out at will. The UAV in the simulation is a hexacopter that is capable of flying in all directions or hover in the same spot.

With the **manual** planning method, the user can fly the UAV around with arrow keys as the clock runs in a sped up fashion. The user can freely switch between two flying modes, turn mode and strafe mode, and four camera views, global view (always north up with full view of the map), behind view, bird's eye, and free form view (where the user can rotate/pan/zoom while flying). The user can pause/resume the flight and path planning to perform the secondary task or just review the search area for better planning.

With the **pattern** planning method, the user can choose from three simple patterns (spiral, lawnmower, and line as shown in the bottom portion of Figure 3) and join these patterns to form the final path. As the user moves the cursor around, the size of the pattern changes with the cursor position marking the end of the path segment (up to the remaining flight time to keep the path valid). Rotation of the lawnmower pattern can be achieved by rotating the map left/right instead. And rotating the map up/down turns the perfect spiral pattern into an ellipse pattern. The user also can undo the last path segment created all the way back to the start. This planning method is "semi-autonomous" because the patterns are generated automatically without manually setting waypoints.

With the **sliding autonomy** method (top portion of Figure 3), the user can (optionally) set an end point anywhere on the map (reachable within remaining flight time) for the current path segment and then see the path suggested by autonomy. Then he/she can drag the knob of the left slider to change the amount of time allocated to autonomy, and see path suggested by autonomy instantly for each time allocation as the knob of the slider is dragged up or down. The slider's max value always reflects the remaining flight time (in minute) for the user to plan. If the user is happy with the current path segment, he/she approves it, the UAV moves to the end of the path segment, and the process repeats until all flight time has been planned. The path planning algorithm used in this planning method is the LHC-GW-CONV algorithm described in [20, 21]. We choose this algorithm because it is the fastest algorithm.

The right slider is used to set the resolution or step value of the left slider and has a range between 1 and 10. For example, if the value of the right slider is set to 10, moving the left slider from bottom up will change time allocation to 10, 20, 30, ..., respectively. The purpose of the second slider is to improve the interaction experience when the user moves the left slider, because in order to provide instant feedback, paths with different time allocation need to be pre-computed by autonomy. Although each path only takes a fraction of a second to generate, for a 60-minute flight autonomy has to generate 60 paths for all possible time allocations, which can take a long time. When the value of the second slider is set high, only a small number of paths need to be pre-computed which enables the instant feedback. This feature turned out to have negative effect on users' interaction experience, which we will discuss later.

With all three planning methods, the user can choose to start over at any time during the exercise, and can restart as many time as exercise time allows, and we record the best path out of all tries. Each user fully understands how the manual and pattern planning methods work, but does not know how path planning autonomy generates paths behind the scene in the sliding autonomy method.

C. Scenarios

The user study contains two WiSAR scenarios. Scenario 1 is a synthetic case with a probability distribution map that is a mixture of five Gaussians (shown in the top portion of Figure 3). No task-difficulty map is used in this scenario (uniform detection probability is assumed).

Scenario 2 comes from a real WiSAR scenario, in which An elderly couple was reported missing near the Grayson Highlands State Park in Virginia. The probability distribution map used for this scenario (Figure 3 middle left) was generated using a Bayesian model [22]. The map has been evaluated at George Mason University's MapScore web portal [44] and performed better than most other models evaluated[3]. This scenario also uses a task-difficulty map, and the map (Figure 3 middle right) was generated using vegetation density data downloaded from the USGS web site and categorized into three difficulty levels (sparse, medium, and dense, with detection probability of 100%, 66.67%, and 33.33% respectively).

Scenario 2 is clearly more complicated than scenario 1 because the user also has to consider the different detection probability defined by the task-difficulty map. We refer to scenario 2 as the high information scenario and scenario 1 as the low information scenario.

D. Secondary Task

In each exercise, we also ask each participant to perform a secondary task together with the main task of path planning. This way we can measure the mental workload of the user. The secondary task is in the form of a group chat window (see the lower left corner of the top picture in Figure 3), and when the user's code name "Eagle" is called, the user is asked to answer simple questions by typing in the chat window. Every 3 seconds (plus a random integer drawn from the uniform distribution [-2,2]) a text message is sent to the chat window, and every 5th message asks the user a simple question. Therefore, every minute the user receives 20 messages and 4 of them are directed to the user. For the same scenario and the same planning method, all users use the same set of chat messages.

We choose to use a group chat window as the secondary task because this is typical in real SAR operations. We also designed the chat messages to simulate a real WiSAR search. The user is asked to acknowledge connection and report path planning status periodically. This design ensures that the secondary task is ecologically valid [45, 46] and makes the experiment result more convincing.

[3]Scoring 0.8184 on a [-1,1] scale where the higher the score the better. http://sarbayes.org/projects/

E. Procedure

Each participant first fills out a demographic survey after signing the IRB consent form, then he/she completes four training exercises. The first three training exercises teach the user how to plan paths with the three planning methods using a simple probability distribution map and no task-difficulty map. The fourth training exercises adds the task-probability map to the path planning problem, and the user gets to practice the manual planning method again. Each training exercise lasts 5 minutes and the user cannot skip it. A "cheat sheet" is provided to each participant during the entire user study to explain the simulation environment and key concepts. Each participant is also asked to read the instructions for the NASA TLX survey.

Then each participant completes the six exercises (2 scenarios and 3 planning methods each) in a counterbalanced order. For each exercise the user has up to 5 minutes to plan a path. Once the user is happy with the path generated, he/she can finish the exercise early. We choose this design because we do not want the user to put all effort into completing the secondary task once he/she considers the primary task completed, which would skew the measurements on secondary task performance.

After the user completes each exercise, we ask the user to complete a NASA TLX survey for the exercise. Then after all six exercises are completed, the user fills out a post user study survey describing his/her subjective preference with the three planning methods.

F. Measures

We use the following five measurements for the primary path planning task:

- **Percent score**: In each exercise, an exercise score is computed by summing the amount of probability collected by the UAV if it followed the path planned. The user's best score for each exercise (because the user can have multiple tries) is normalized by dividing the best score from all users for the same scenario to compute the percent score. This way we can compare planning methods across scenarios.
- **Time spent**: How much time is spent with each exercise.
- **Try count**: How many times the user tried in each exercise. Note that because the manual planning method takes much longer time to plan a path than the other two methods by design, this measurement is used mainly to compare between the pattern and sliding autonomy planning methods.
- **Mouse clicks per try**: How many times the user left-clicked the mouse within a try. Again, this measurement is used to compare pattern and sliding autonomy planning methods because manual planning methods does not require a lot of mouse clicks by design.
- **NASA-TLX raw score**: The sum of user subjective evaluation of cognitive workload in six dimensions normalized to a 100-point scale.

The following two measurements are used for the secondary task:

- **Percent of questions missed**: What percentage of questions directed to the user were missed before user completed the exercise. Here we do not measure the percent of questions answered correctly because all the questions are very simple and all users answered the questions correctly.
- **Chat latency**: The number of seconds between the time a question was presented to the user and the time when the user answered the question.

VI. Results and Analysis

We analyzed data recorded with a repeated measure ANOVA and report results in this section.

A. Compare Across Scenarios

During the user study, each participant worked through two WiSAR scenarios. In scenario 1 (low information) no task-difficulty map is used. In scenario 2 (high information) because partial detection is enforced, the amount of probability that can be collected (exercise score) in scenario 2 is much lower than in scenario 1. In order to show a fair comparison across scenarios, we use percent score instead, which is a percent comparing the participant's score to the best performing participant's score in the same scenario.

Table I lists the user study results compared between scenario 1 and 2 with statistically significant results highlighted in bold. Average participant percent score is very close in both scenarios because we normalized exercise scores. So an average participant's performance against the highest score is about 75% in each scenario. There is a slight difference of 26 seconds in average time spent, which is not significant compared to the 300 seconds given for each exercise. Average try count in both scenarios are close enough, meaning each participant had enough time in each exercise to give each scenario a few tries.

Mouse clicks per try for the two scenarios are significantly different ($F = 28.65$, $p < .0001$) indicating scenario 2 required more physical work from each participant than scenario 1. This result matches the difficulty of the two scenarios where each participant created more path segments with the pattern and sliding autonomy planning methods in scenario 2 because it uses a task-difficulty map with many "valleys".

NASA TLX score is also significantly different ($F = 31.35$, $p < .0001$) between the two scenarios where the average score difference is 9.98 (out of a total of 100 points), almost a full "pip" on the TLX survey, indicating that on average each participant felt his/her cognitive workload was much higher in the high information scenario.

The percent of questions missed is only slightly different between scenarios (52.94% and 56.69%), and the chat latency is also very close (10.39 and 11.17 seconds). This shows that participants performed about the same on average with the secondary task across scenarios.

If we only look at the performance of the sliding autonomy planning method across scenarios, results follow the same trend. Therefore we do not list the results separately.

TABLE I
Comparing across scenarios

	S1 Low	S2 High	SE	Significance
Score %	76.99	74.17	1.12	F=7.51, p=.0112
Time spent	224.01	250.47	12.06	F=8.35, p=.0079
Try count	3.08	2.67	0.37	F=3.32, p=.804
Clicks/try	15.95	33.54	2.59	**F=28.65, p <.0001**
NASA TLX	48.19	58.17	2.50	**F=31.35, p <.0001**
Q. missed %	54.88	54.90	5.18	F=0.00, p=.9941
Chat latency	10.88	10.77	0.56	F=0.03, p=.8755

TABLE II
Comparing across planning methods

	M	P	SA	SE	Significance
Score %	59.40	72.75	94.60	1.39	**F=223.03, p <.0001**
Time spent	243.35	240.02	228.37	12.06	F=1.16, p=.3226
Try count	1.75	3.56	3.31	0.43	F=9.47, p=.0003
Clicks/try	13.01	35.64	25.58	2.90	**F=19.47, p <.0001**
NASA TLX	61.51	49.18	48.86	2.81	**F=14.15, p <.0001**
Q. missed %	52.94	56.69	55.04	5.17	F=1.26, p=.2915
Chat latency	10.39	11.17	10.92	0.65	F=0.46, p=.6327

B. Compare Across Planning Methods

For each scenario, three path planning methods were used. Table II lists comparison among the three methods.

The performance difference in percent score is statistically significant ($F = 223.03$, $p < .0001$) with pattern (72.75%) performing better than manual (59.40%) and sliding autonomy (94.60%) performing better than pattern. As shown in Figure 4, this trend is also clear in both scenario 1 and 2. Therefore, user study results show strong support for our first hypothesis: sliding autonomy method performs better than either the manual method or the pattern method in both low information and high information scenarios.

Statistically significant difference was also found in mouse clicks per try (F =19.47, p <.0001). The manual method uses arrow keys to fly the UAV around and only uses mouse clicks when switching camera modes or stop the timer in order to perform the secondary task. Therefore, by design, this method does not use a lot of mouse clicks. Pattern and sliding autonomy methods both use mouse clicks for the actual path planning task, and the pattern method clearly generated more mouse clicks per try (35.64) than the sliding autonomy method (25.58). Two factors might have contributed to this difference,

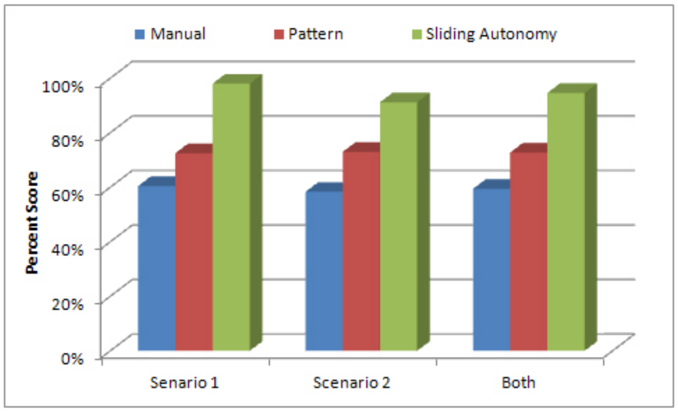


Fig. 4. Performance difference for three path planning methods.

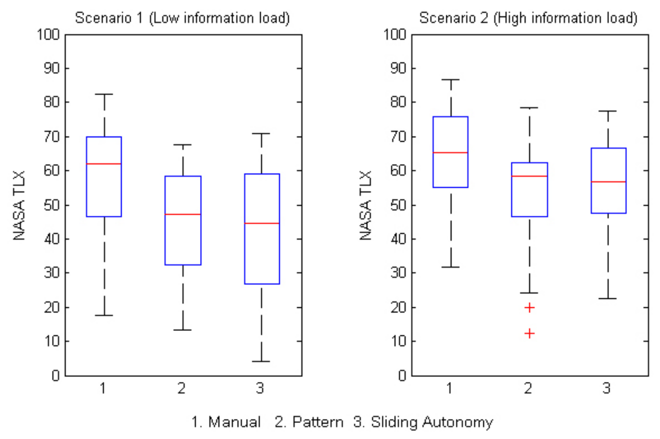


Fig. 5. Box plots of the NASA TLX scores for each scenario.

first one being that the pattern method allows undoes while the sliding autonomy method does not support this function. Also with the sliding autonomy method a participant can drag the slider and see different suggestions from path planning autonomy, which does not require a lot of mouse clicks.

NASA TLX raw scores show significant differences among the three methods (F =14.15, p <.0001), with the manual method showing the highest cognitive mental workload (61.51), a full "pip" more than the other two methods. However, the score difference between the pattern method and the sliding autonomy method is small. Figure 5 shows the box plots of the NASA TLX scores for each scenario. Participants felt that the pattern method and the sliding autonomy method required about the same amount of cognitive mental workload in each scenario, and the manual method required a lot more mental workload in both scenarios.

For all three planning methods, participants performed about the same on the secondary task, as shown by percent of questions missed and chat latency in Table II. Combining this with percent score and NASA TLX we can conclude that sliding autonomy performed best without increasing participants' mental workload, which support our third hypothesis: Sliding autonomy method does not increase the mental workload of the operator when compared against manual and pattern methods.

If a participant allocates the entire flight time (60 minutes) to path planning autonomy (LHC-GW-CONV algorithm), the path is generated by full autonomy without any human input. The percent score for this path would be 96.13% for scenario 1 and 78.83% for scenario 2. This is better than the average performance of the manual and pattern methods in both scenarios as shown in Figure 6. If we compare each person's percent score against full autonomy out of all 26 participants, in scenario 1, none could outperform full autonomy using the manual and pattern methods, but 23 (88.46%) were able to perform better with sliding autonomy (human-autonomy team). In scenario 2, no one did better using the manual method, 5 (19.23%) did outperform full autonomy using the pattern method, and 24 (92.31%) were able to beat full autonomy using the sliding autonomy method. Table III summarizes the

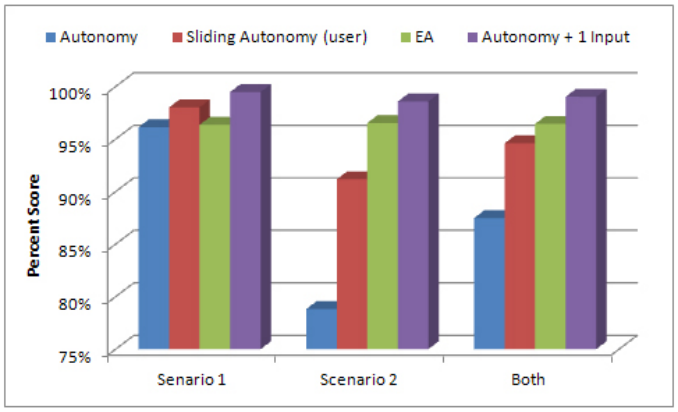


Fig. 6. Comparing sliding autonomy performance against various markers.

TABLE III
PERCENT OF PARTICIPANTS OUTPERFORMING AUTONOMY WITH EACH METHOD

	Manual	Pattern	Sliding Autonomy
Scenario 1 (Low)	0%	0%	**88.46%**
Scenario 2 (High)	0%	19.23%	**92.31%**

comparison. User study results clearly show that the human inputs made significant differences and human-autonomy team outperformed autonomy working alone. This also strongly supports our second hypothesis: Sliding autonomy method performs better than autonomy working alone in both low information and high information scenarios.

C. Additional Factors

We also performed ANOVA analysis on some additional factors that might create differences: gender, experience in video games, order of the scenarios, and whether participants used full autonomy with the sliding autonomy method. No significant differences were found for these factors overall, across scenarios, or across methods. Table IV below lists analysis results. There is also no significant correlation (-0.2259) between percent of questions missed in the secondary task and the NASA TLX raw scores.

VII. DISCUSSION

A. Planning methods looking more closely

With the manual method, the user uses arrow keys to move the UAV around to create a path, so by design the method is very intuitive, flexible, and requires more physical interactions (keyboard, not mouse clicks). But in order to plan a 60-minute path in 2 minutes, the UAV has to move very quickly in the simulation environment. We made sure each user can

TABLE IV
ANOVA ANALYSIS RESULTS FOR ADDITIONAL FACTORS

	Overall	Scenario	Method
Gender	F=0.05, p=.8292	F=0.76, p=.3930	F=0.59, p=.5602
Video Game Exp.	F=0.78, p=.5497	F=1.64, p=.2011	F=1.13, p=.3650
Scenario Order	F=0.09, p=.7672	F=0.39, p=.5378	F=1.53, p=.2278
Full Autonomy	F=3.70, p=.0675	F=0.36, p=.5559	F=0.04, p=.9599

complete at least two tries using this method while performing the secondary task simultaneously. High UAV speed makes it harder to navigate accurately. Many participants called the arrow keys too "sensitive" and recommended slowing down the UAV. Because of the time pressure, the path planning task is more of a continuous process. Therefore, when errors are made, it is too costly to start over (we did not provide an undo function). The user also does not have the luxury to pause the planning in order to think through strategies in his/her head; it has to be done while the user is moving the UAV around. Naturally, when this continuous process is frequently interrupted by the secondary task where the user has to pause planning and answer questions in the group chat window, user frustration goes high. More physical work, higher frustration, and low performance score are the main factors contributing to a much higher NASA TLX score for the manual method. Although this planning method supports four camera views and two flying modes, most participants chose to stick with free form view where the entire search region is displayed and with strafe mode to avoid getting confused with UAV orientation. During training, participants actually had one extra exercise with the manual method, but this method still performed the worst.

With the pattern method, the user joins a mixture of three patterns (spiral, lawnmower, and line) together to form the final path. This is more of an episodic process, so it is very easy to pause in the middle of the planning and shift attention to the secondary task. There is also less time pressure because the user can quickly plan for the remaining time with just one big spiral (or lawnmower) in one click. Therefore, the user has plenty of time to try many times with different strategies. In the post user study survey, many participants commented that with the spiral and lawnmower pattern it is really easy to run out of time. They suggested adding the ability to allocate time to the patterns similar to the sliding autonomy method. This means that with this method, a user enjoys the systematic coverage of an area but has a hard time estimating how much time it takes the UAV to cover the area following the pattern. Several participants also suggested adding more patterns to the method. The pattern method is the only method supporting undoes. This ability surely had impacts on number of mouse clicks per try and participants' preference over the three planning methods. Another interesting observation is that participants seemed to be overly optimistic about their performance using the pattern method. For example, although sliding autonomy performed better than pattern in all scenarios for all participants, in 46.15% of the times, participants rated pattern as performing better than sliding autonomy in NASA TLX scores; and also in the post user study survey, 26.92% participants (not the majority, but over a quarter of the population) actually believed the pattern method creates best paths.

Similar to the pattern method, the sliding autonomy method is also an episodic process. Therefore, stopping in the middle of the planning to answer questions for the secondary task was easy. And it only takes a few clicks to let full autonomy plan path for the remaining time, so there is not much time pressure and the user can have many tries. Because the user does not know how autonomy works behind the scene, many participants were surprised by the path recommended by autonomy, and feel that autonomy did not do what they had tell it to do. For example, when a user sets the end point in region A, autonomy might plan a path that spends most of time in a seemingly unrelated region B and only goes toward region B at the end of the path, because such a path is more efficient (scores higher). In such cases, the slider becomes the only tool that lets the user "force" autonomy to do what the user wants, and path planning turns into a fight between the user and autonomy. However, the instant feedback (displaying path and the predicted "vacuuming effect") does help the user figure out why autonomy would suggest something different, and some participants were glad that autonomy suggested better paths they had not considered. Most participants were generally happy with the path segment recommended by autonomy covering a local region, even when the region is in an irregular shape (not a circle or rectangle). Many participants also expressed that they did not have enough control over the path generation and recommended adding the ability to include constraints such as "middle points" where the path segment has to go through these middle points. In fact, such "middle points" can already be achieved with the current method by setting end points. Several participants complained that this method does not have the undo function. They also wanted the ability to see how suggested path changes when the end point is moved around, which is an idea worth exploring because it allows the user to not only slide along the time allocation dimension, but also slide along the constraints dimension and see instant feedback. With both the pattern and sliding autonomy methods, many participants expressed the desire to be able to modify the path after it is generated.

In scenario 2 where a task-difficulty map was used, because the probability map is similar to a unimodal distribution (see middle portion of Figure 3, most participants chose to first cover the probability hill use the probability distribution map, and then planned the remaining path using only the task-difficulty map view in all three methods. If the probability distribution map is more complex with many probability hills, then likely the user will be switching between the two maps back and forth in order to incorporate information. Some suggested showing both maps side by side or have a way to combine the two maps into one. These ideas are worth exploring in future user studies.

In the post user study survey, the majority of the participants think manual is the easiest to learn (53.85%), pattern is the easiest to use (57.69%), and sliding autonomy performed the best (65.38%). However, most participants preferred the pattern method (69.23%) out of all three. We believe the inability to undo, not able to move the end point once path is suggested, and the second slider in the interface (which we will discuss in a later section) all had negetive impacts on participants' preference over the sliding autonomy method. Once these functions are in place, the sliding autonomy method will become more preferrable than the pattern method.

B. Trust in autonomy

The algorithm we used for the sliding autonomy method is the LHC-GW-CONV algorithm. We selected this algorithm because of its fast speed. If speed is not a concern, then the Evolution Algorithm (EA) we developed [20] can autonomously generate even better paths. For example, the percent score for EA would be 96.36% for scenario 1 and 96.54% for scenario 2. In scenario 1, the score is only slighly better than full autonomy in sliding autonomy (96.13%), but in scenario 2, the score is much better than full autonomy (78.83%). We also used the sliding autononomy method and generated paths for both scenario 1 and 2 with only one human input, an end point. Using the best score out of 3 tries (roughly equal to the average number of tries in the user study), we also computed the percent score for this autonomy+1 human input approach: 99.47% for scenario 1 and 98.58% for scenario 2. Using the EA and Autonomy+1 score as additional markers, we plotted participants average performance in each scenario against these markers. Figure 6 shows the result.

First, sliding autonomy (human-autonomy team) outperformed autonomy (LHC-GW-CONV algorithm) in both scenarios. Sliding autonomy also outperformed EA in scenario 1 (low information). In scenario 2, the performance of sliding autonomy is not very far from EA (5.36%), and the difference is even smaller (1.85%) when averaged over both scenarios. However, the most interesting observation is that autonomy+1 actually outperformed all others in both scenarios (99.47% for scenario 1 and 98.58% for scenario 2). Although a few participants did score higher than autonomy+1, the difference is less than 1.5%. Table V lists what percentage of participants outperformed full autonomy, EA, and autonomy+1.

What Figure 6 suggests is that with the sliding autonomy method, it does not need a lot of human inputs to perform really well. Instead of spending the effort creating many path segments and set many end points, it is probably more rewarding to search for the right region to set just one or two constraints. However, 88.68% of participants gave more than 1 input when they used sliding autonomy (81.13% for 2 inputs and 69.81% for 3 inputs). This suggests that human users lacked trust for the performance of autonomy. In the post user study survey, only 46.15% participants acknowledged that he/she used full autonomy during path planning with the sliding autonomy method. This number also indicates that most participants did not trust autonomy. When using the sliding autonomy method, a good strategy is actually to start with full autonomy (as the worst performance base) and then see how additional human inputs can improve the path.

Lee and See propose that because people respond to technology socially, trust guides reliance when unanticipated situations make it impractical or impossible to understand automation [47], and defined *distrust*, "trust falls short of the automation's capabilities", and *overtrust*, "trust exceeds system capabilities". Bradshaw et al. [7] had similar definitions and used the term *under-reliance* for distrust, instead. Both suggested that trust in automation should be "calibrated". Hoffman et al. [48] suggests "active exploration for trusting"(AET) and hope this approach can promote both trust "calibration" and appropriate reliance. We believe the sliding autonomy approach we propose can also be used for trust calibration. As the user is dragging the knob on the slider, he/she is calibrating his/her trust on path planning autonomy.

TABLE V
PERCENT OF PARTICIPANTS OUTPERFORMING AUTONOMY PERFORMANCE MARKERS

	Autonomy	EA	Autonomy+1
Scenario 1 (Low)	**88.46%**	**88.46%**	7.69%
Scenario 2 (High)	**92.31%**	26.92%	15.38%

In our user study, all participants only had 30 minutes of training before using the sliding autonomy method. We speculate that as the user gets more familiar with the sliding autonomy approach in the long run, he/she would be able to calibrate trust better. However, validation of this claim is beyond the scope of this paper and will be part of a future long-term user study.

When more complicated probability distribution map and task-difficulty map are used, it is possible that the human user will see more value in the sliding autonomy approach because combining this much information in human mind and then decide trade offs in path planning required much more computation. And when the human user finds himself/herself incapable of such tasks, he/she might rely on autonomy to help solve the puzzle, and the sliding autonomy approach becomes more useful in calibrating trust. Validating this hypothesis is another natural extension of the present work.

C. *Why human-autonomy team performs better?*

User study results show that the human-autonomy team outperformed either human or autonomy working along. But how were they able to achieve this? We believe that the reason is: the sliding autonomy approach enabled the human to focus on what human is good at and autonomy to focus on what autonomy is good at. Bradshaw et al. pointed out in [7]: "Humans, though fallible, are functionally rich in reasoning strategies and their powers of observation, learning, and sensitivity to context." Our observation suggests that human is really good on two things: planning strategically and recognize bad path segments.

The sliding autonomy method lets the user plan at a higher abstract level by specifying priorities in search sub-regions and how well each sub-region should be covered. Then how to cover the sub-region well given a fixed flight time is left for autonomy to handle. Autonomy, on the other hand can generate a path that covers a sub-region (or some nearby sub-regions) precisely and quickly, and can handle all kinds of irregular sub-region shapes. Therefore, the sliding autonomy method combines the strengths of both human and autonomy.

A human user is also very good at recognizing bad moves in solutions suggested by autonomy. The sliding autonomy approach enables the human user to select from a bunch of suggested paths, selecting a path segment with fewer bad moves (which might not actually be bad in autonomy's perspective with just the current path segment) will probably increase the chance of a good final path. So again, the sliding approach takes advantage of the strengths of both human and autonomy.

D. Why similar secondary task performance in all three methods?

The pattern and sliding autonomy methods are episodic, suggesting that it is easier for the user to pause planning and shift attention to the secondary task of answering questions in the group chat window. It almost seems a natural conclusion that the user should have performed better with the secondary task compared to the manual method. However, user study data shows that there is no significant differences in the users secondary task performance across all three path planning methods.

The manual method required a lot of continuous keyboard interaction (great physical demand and temporal demand) to move the UAV around. However, it does not actually require a lot of mental demand and effort because in the user's mind he/she is not actually planning things in great detail. If a mistake is made in the path planning, because there is no way to correct it, the user quickly drops it out of his/her mind and stop worrying about it. The planning process is more sporadic and spontaneous. Therefore, the low mental demand and effort makes monitoring the group chat window an easy task, even though switching back between primary task and secondary task is very frustrating.

With the pattern and sliding autonomy methods, path planning is more like piecing together a puzzle. The user is deeply drawn into the problem solving mode, and has to consider many trade offs in his/her mind, which actually required more mental involvement. With the sliding autonomy method, the user is interacting with complicated algorithms, so while planning a path, the user is also trying to build a mental model of how autonomy works. Therefore, the user actually paid less attention to the secondary task. Fighting with autonomy when human and autonomy had disagreements and the user's struggle with the second slider also drew user attention away from the group chat window. But when the group chat window catches the user's attention, he/she can perform the secondary task leisurely.

Another factor affecting the secondary task performance is summarized by David Woods and Eric Hollnagel as the law of stretched systems [6]: ”every system is stretched to operate at its capacity; as soon as there is some improvement, for example in the form of new technology, it will be exploited to achieve a new intensity and tempo of activity.” Therefore, with the pattern and sliding autonomy methods, people had more tries and evaluated more options and trade offs. With the sliding autonomy method, this means the user played with more time allocations and evaluated more paths suggested by path planning autonomy, which ultimately resulted in better quality paths at the cost of not so great performance in the secondary task.

E. Fighting with Human Nature

It seems buried deeply in human nature that when a user performs an action, he/she wants immediate feedback so badly that when no such feedback is available, he/she cannot fight the urge to perform more actions thinking the additional actions would generate some feedback. To meet this demand, when we designed the sliding autonomy interface, we implemented two sliders where the second slider sets the step size value of the first slider. We had hoped that this would reduce the number of paths we had to pre-compute and make the user experience smoother by providing instant path feedback when they move the first slider. Observations from the user study show that this design actually negatively affected the user experience and increased user's cognitive workload. Which also means a better design with this element could potentially improve the user's performance and preference with the sliding autonomy method.

Because when the value of the second slider is changed, the value of the first slider is also changed to reflect the smallest number that is a multiple of the selected step value. This behavior created multiple problems. First, some users quickly learned that changing the value of the second slider also changes time allocation to the path planning task, and started using this slider to set time allocation instead. This meant that more paths actually had to be pre-computed behind the scene, and in the user's eyes now the system was no longer providing instant feedback. So he/she began moving the slider randomly with the hope to generate some feedback, which meant more paths had to be pre-computed. A very bad cycle. Secondly, even though during training each user is told that there might be slight delay when they move the main slider in some cases, in the real exercises, they still wanted instant feedback. And when instant feedback was not available, they fell in the trap of control oscillation, and had to make several tries to set the value they desire. Fighting with the sliders required additional attention, so less attention were paid to monitor the group chat window. A “please wait” message would probably have less negative impact on the user's experience.

When a human sees an “obvious error” in a path segment, the urge to correct is typically so strong that it draws all the attention to the quest, turning the path planning task into a fight with autonomy at the cost of increased cognitive workload. The “obvious error” could mean two different cases. In the first case, the “error” is indeed a bad move that can be easily modified in the user's mind to improve the path performance. In the second case, the “error” is actually not a bad move, only that the user cannot comprehend the reason behind the move, so there is still a strong desire to change it. However, because there is no direct way for the user to modify the path, the user can only change the path by moving the slider to set different time allocation, sometimes this “error” simply cannot be modified. The user wastes time trying, user's cognitive workload increases, and the user has a bad experience with the sliding autonomy method.

There are probably more things than what we discussed here that could have improved the user experience and improve the user's performance on both the primary task of path planning and the secondary task of answering questions in the group chat window. We just want to emphasize that even with these negative impacts, the sliding autonomy method still performed significantly better than the other two methods and autonomy working alone without increasing the user's mental workload.

VIII. Conclusions and Future Work

In this paper we propose a new autonomy management approach, sliding autonomy, which lets the user influence the behavior of the autonomous system along three new dimensions: information representation, task constraints, and time allocation. We extend the autonomy design guideline in our prior work by adding a new row for intelligence of collaborative agents (including human-autonomy team), and explain how the three new dimensions fit into the guideline when we apply the proposed approach to the task of UAV (Unmanned Aerial Vehicle) path planning to support Wilderness Search and Rescue (WiSAR). Experiment results from a user study validate our hypotheses and show that the sliding autonomy method performs significantly better than either the manual or pattern path planning method without increasing the user's mental workload, and the human-autonomy team outperforms either human or autonomy working alone.

Possible future work include evaluating the sliding autonomy approach in a long-term study and see how the user's trust can be calibrated. More complicated WiSAR scenarios can be tested and see how that affects the human-autonomy interaction and the performance of the human-autonomy team. Another user study can evaluate how well the human-autonomy team can work with outdated information representation using the sliding autonomy approach. It would also be interesting to investigate the human-autonomy interaction experience when the ability to modify information representation is added to the tool.

Acknowledgments

This work was partially supported by the National Science Foundation under grant number 0534736 and by a grant from the Army Research Laboratory. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsoring organizations.

References

[1] A. Chun, "Optimizing Limousine Service with AI," *Proceedings of the Twenty-Second Innovative Applications of Artificial Intelligence Conference (IAAI-10)*, 2010.

[2] J. Casper and R. Murphy, "Human-Robot Interactions During the Robot-Assisted Urban Search and Rescue Response at the World Trade Center," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 33, no. 3, pp. 367–385, 2003.

[3] L. Lin, M. Roscheck, M. A. Goodrich, and B. S. Morse, "Supporting Wilderness Search and Rescue with Integrated Intelligence: Autonomy and Information at the Right Time and the Right Place," in *Procedings of Twenty-Fourth AAAI Conference on Artificial Intelligence, Special Track on Integrated Intelligence.*, 2010.

[4] B. Robins, K. Dautenhahn, and P. Dickerson, "From Isolation to Communication: A Case Study Evaluation of Robot Assisted Play for Children with Autism with a Minimally Expressive Humanoid Robot," in *Advances in Computer-Human Interactions, 2009. ACHI'09. Second International Conferences on*, feb. 2009, pp. 205 –211.

[5] L. Bainbridge, "Ironies of Automation." *Automatica*, vol. 19, no. 6, pp. 775–780, 1983.

[6] D. D. Woods and E. Hollnagel, *Joint cognitive systems: Patterns in cognitive systems engineering*. CRC Press, 2006.

[7] J. M. Bradshaw, R. R. Hoffman, M. Johnson, and D. D. Woods, "The Seven Deadly Myths of Autonomous Systems," *Intelligent Systems, IEEE*, vol. 28, no. 3, pp. 54–61, 2013.

[8] T. Sheridan, *Telerobotics, automation, and human supervisory control*. The MIT press, 1992.

[9] M. Hearst, "Mixed-Initiative Interaction," *IEEE Intelligent systems*, vol. 14, no. 5, pp. 14–23, 1999.

[10] T. Fong, C. Thorpe, and C. Baur, "Collaborative Control: A Robot-Centric Model for Vehicle Teleoperation," in *AAAI 1999 Spring Symposium: Agents with Adjustable Autonomy*, 1999.

[11] G. A. Dorais, R. P. Bonasso, D. Kortenkamp, and B. P. D. Schrechenghost, "Adjustable Autonomy for Human-Centered Autonomous Systems on Mars," in *The First International Conference of the Mars Society*, 1998.

[12] G. Dorais and D. Kortenkamp, "Designing Human-Centered Autonomous Agents," in *Advances in Artificial Intelligence. PRICAI 2000 Workshop Reader*. Springer, 2001, pp. 321–324.

[13] M. Dias, B. Kannan, B. Browning, E. Jones, B. Argall, M. Dias, M. Zinck, M. Veloso, and A. Stentz, "Sliding Autonomy for Peer-to-Peer Human-Robot Teams," *Intelligent Autonomous Systems 10: IAS-10*, p. 332, 2008.

[14] W. Rouse, "Adaptive Aiding for Human/Computer Control," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 30, no. 4, pp. 431–443, 1988.

[15] D. Kaber, J. Riley, K. Tan, and M. Endsley, "On the Design of Adaptive Automation for Complex Systems," *International Journal of Cognitive Ergonomics*, vol. 5, no. 1, pp. 37–57, 2001.

[16] J. Bradshaw, P. Feltovich, H. Jung, S. Kulkarni, W. Taysom, and A. Uszok, "Dimensions of Adjustable Autonomy and Mixed-Initiative Interaction," *Agents and Computational Autonomy*, pp. 235–268, 2004.

[17] R. Murphy, E. Steimle, C. Griffin, C. Cullins, M. Hall, and K. Pratt, "Cooperative Use of Unmanned Sea Surface and Micro Aerial Vehicles at Hurricane Wilma," *Journal of Field Robotics*, vol. 25, no. 3, pp. 164–180, 2008.

[18] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L.Cooper, M. Quigley, J. A. Adams, and C. M. Humphrey, "Supporting Wilderness Search and Rescue using a Camera-Equipped Mini UAV," *Journal of Field Robotics*, vol. 25, no. 1-2, pp. 89–110, 2008. [Online]. Available: http://www3.interscience.wiley.com/cgi-bin/fulltext/117865012/PDFSTART

[19] F. Bourgault, T. Furukawa, and H. Durrant-Whyte, "Coordinated decentralized search for a lost target in a Bayesian world," in *Proc. IEEE/RSJ Int. Conf. IROS*, 2003.

[20] L. Lin and M. A. Goodrich, "UAV Intelligent Path Planning for Wilderness Search and Rescue," in *Proceedings of IEEE/RSJ Int. Conf. IROS*. IEEE, Oct 2009, pp. 709–714.

[21] ——, "Hierarchical Heuristic Search Using A Gaussian Mixture Model for UAV Coverage Planning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2014, submitted, under 2nd round of review.

[22] ——, "A Bayesian approach to modeling lost person behaviors based on terrain features in Wilderness Search and Rescue," *Computational & Mathematical Organization Theory*, vol. 16, no. 3, pp. 300–323, Sep 2010.

[23] S. Clark and M. A. Goodrich, "A hierarchical flight planner for sensor-driven UAV missions," in *RO-MAN, 2013 IEEE*. IEEE, 2013, pp. 509–514.

[24] P. F. Drucker, *The Practice of Management*. Harper Paperbacks, Oct. 2006.

[25] M. Goodrich and A. Schultz, "Human-Robot Interaction: A Survey," *Foundations and Trends in Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007.

[26] T. Sheridan and W. Verplank, "Human and Computer Control of Undersea Teleoperators," 1978.

[27] R. Parasuraman, T. Sheridan, and C. Wickens, "A Model for Types and Levels of Human Interaction with Automation," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000.

[28] J. Bradshaw, M. Sierhuis, A. Acquisti, P. Feltovich, R. Hoffman, R. Jeffers, D. Prescott, N. Suri, A. Uszok, and R. Van Hoof, "Adjustable Autonomy and Human-Agent Teamwork in Practice: An Interim Report on Space Applications," *Agent Autonomy*, pp. 243–280, 2003.

[29] D. Kaber, M. Wright, L. Prinzel III, and M. Clamann, "Adaptive Automation of Human-Machine System Information-Processing Functions," *Human factors*, vol. 47, no. 4, p. 730, 2005.

[30] J. Brookshire, S. Singh, and R. Simmons, "Preliminary Results in Sliding Autonomy for Coordinated Teams," in *Proceedings of The 2004 Spring Symposium Series*, 2004.

[31] N. Sarter, "Making Coordination Effortless and Invisible: The Exploration of Automation Management Strategies and Implementations," in *CBR Workshop on Human Interaction with Automated Systems*, 1998.

[32] D. Norman, *Some Observations on Mental Models*. Lawrence Erlbaum, 1983.

[33] M. Quigley, B. Barber, S. Griffiths, and M. A. Goodrich, "Towards real-world searching with fixed-wing mini-UAVs," in *Proceedings of IROS*, 2005.

[34] J. K. Howlett, T. W. McLain, and M. A. Goodrich, "Learning Real-Time A* Path Planner for Unmanned Air VehicleTarget Sensing," *Journal of Aerospace Computing, Information, and Communication*, vol. 3, no. 3, pp. 108–122, 2006.

[35] A. Stentz, *Optimal and efficient path planning for partially known environments*. Springer US, 1997, ch. chapter 11, pp. 203–220.

[36] S. A. Bortoff, "Path planning for UAVs," in *American Control Conference, 2000. Proceedings of the 2000*, vol. 1. IEEE, 2000, pp. 364–368.

[37] R. Beard, D. Kingston, M. Quigley, D. Snyder, R. Christiansen, W. Johnson, T. McLain, and M. Goodrich, "Autonomous Vehicle Technologies for Small Fixed-Wing UAVs," *AIAA Journal of Aerospace Computing, Information, and Communication*, vol. 2, no. 1, pp. 92–108, 2005.

[38] P. O. Pettersson and P. Doherty, "Probabilistic roadmap based path planning for an autonomous unmanned helicopter," *Journal of Intelligent and Fuzzy Systems*, vol. 17, no. 4, pp. 395–405, 2006.

[39] R. C. Holte, M. B. Perez, R. M. Zimmer, and A. J. MacDonald, "Hierarchical A*: Searching abstraction hierarchies efficiently," in *AAAI/IAAI, Vol. 1*. Citeseer, 1996, pp. 530–535.

[40] N. Meuleau and R. I. Brafman, "Hierarchical Heuristic Forward Search in Stochastic Domains," in *IJCAI*, vol. 7, 2007, p. 2542.

[41] M. Naveed, D. E. Kitchin, and A. Crampton, "A Hierarchical Task Network Planner for Pathfinding in Real-Time Strategy Games," in *Proceedings of the Third International Symposium on AI & Games, Daniela M. Romano and David C. Moffat (Eds.),*, 2010, pp. 1–7.

[42] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte, *Optimal Search for a Lost Target in a Bayesian World*, ser. Springer Tracts in Advanced Robotics. Springer-Verlag, 2006, vol. 24, ch. chapter 21, pp. 209–222.

[43] F. Bourgault, A. Goktogan, T. Furukawa, and H. Durrant-Whyte, "Coordinated Search for a Lost Target in a Bayesian World," *Advanced Robotics*, vol. 18, no. 10, pp. 979–1000, 2004.

[44] E. Cawi, N. Jones, and C. R. Twardy, "MapScore: Probability Map Evaluation for Search & Rescue," in *Virginia Search & Rescue Conference*. Conference Presentation presented at the Virginia Search & Rescue Conference, Holiday Lake, VA, 2012.

[45] K. Vicente, "Should an Interface Always Match the Operators Mental Model?" *Cseriac Gateway*, vol. 8, no. 1, pp. 1–5, 1997.

[46] J. Rasmussen, A. Pejtersen, and L. Goodstein, *Cognitive Systems Engineering*. Wiley-Interscience, 1994.

[47] J. Lee and K. See, "Trust in Automation: Designing for Appropriate Reliance," *Human factors*, vol. 46, no. 1, p. 50, 2004.

[48] R. R. Hoffman, M. Johnson, J. M. Bradshaw, and A. Underbrink, "Trust in Automation," *Intelligent Systems, IEEE*, vol. 28, no. 1, pp. 84–88, 2013.