2015 IEEE International Conference on Systems, Man, and Cybernetics

Human-assisted RRT for Path Planning in Urban Environments

S. S. Mehta[a†], C. Ton[a], M. McCourt[a], Z. Kan[a], E. A. Doucette[b], W. Curtis[b]
[a]Research and Engineering Education Facility, University of Florida, Shalimar, FL-32579, USA
[b]Munitions Directorate, Air Force Research Laboratory, Eglin AFB, FL-32542, USA

***Abstract*—A human-RRT (Rapidly-exploring Random Tree) collaborative algorithm is presented for path planning in urban environments. The well-known RRT algorithm is modified for efficient planning in cluttered, yet structured urban environments. To engage the expert human knowledge in dynamic replanning of autonomous vehicles, a graphical user interface is developed that enables interaction with the automated RRT planner in real-time. The interface can be used to invoke standard planning attributes such as wayareas, space constrains, and waypoints. In addition, the human can draw desired trajectories using the touch interface for the RRT planner to follow. Based on new information and evidence collected by human, state-dependent risk or penalty to grow paths based on an objective function can also be specified using the interface.**

I. Introduction

The performance of autonomous systems in complex environments can be improved by including human perceptions and expert knowledge. Within a system, automation and humans may be allocated with different tasks based on cognitive workload, capability or expertise, execution speed, etc. demanded by the tasks [1]. Alternatively, automation working with humans may reduce cognitive burden while benefiting from valuable inputs from humans, which may include recalled facts, measurements fused from heterogeneous sources including soft information from human observers [2], [3], and decisions made in response to new and undocumented situations wherein robots often fall short. Clearly, there is some value to be gained by including human observations and decisions [4], [5], which leads to an important question of how to combine automation and human control for the most effective performance [6], [7].

Path planning is a fundamental problem in robotics with application in robotic exploration, target tracking, autonomous guidance, etc., that aims at finding a feasible path to a goal location based on an objective function and a set of constraints. The automated path planning algorithms can be classified into the following categories: grid-based search, interval-based search, geometric algorithms, potential fields, and the state-of-the-art sampling-based algorithms. Although the automated algorithms work well in many cases, they are known to have limitations in complex environments and may fail to find a solution, get trapped in local minima, obstacles, or map dead-ends, or require excessive computation time and resources. While automated algorithms are preferred, especially in multi-agent systems [8], due to computational advantages, given the brittleness of automation to unforeseen and untaught scenarios, human involvement in path planning becomes critical. Humans with expert knowledge, a result of technical practices, training, and experience and reasoning skills, can overcome the limitations of automation when working in collaboration with the planning algorithms.

Within the robotics community, there is increased momentum in exploring ways to include human-generated information and reasoning into the automated path planning operation. The objective of human collaboration is largely to inject optimality into the generated solution or to affect the search process of the automated planning algorithms by defining goals or objective functions [9]–[16]. The high-level course of action and goals can be defined by an operator for the automated algorithm to generate feasible paths, which can be subsequently reviewed, approved, and executed by an operator (*cf.* [9], [10], [13], [15]). A dynamic environment can be characterized by varying situation awareness, objective function, and scene features (e.g., obstructions, hazards). To improve the quality and suitability of the paths in dynamic environments, the expert knowledge should be exploited by engaging the operator in close collaboration with the automated planners. The operator may interact with the planner to alter the automation-generated path, for example, by moving, adding, or removing waypoints [11], [12], [14], [17]–[20], constraining the planning space or delimiting the admissible space [20], [21], or modifying the parameters of the objective function [22]. In a separate study, Clare et al. [23] assessed the operator workload and system performance as a result of replan prompting rate, i.e., when the operator chooses to compare the automation-generated path with the original path. In [24], an uncertainty reasoning based model is proposed to incorporate human reasoning and perceptions into the automated algorithm. Rapidly-exploring Random Tree (RRT) [25] is a well-known sampling-based path planning algorithm suitable for higher-dimensional spaces. The convergence and coverage properties of the RRT algorithm can be improved with appropriate human collaboration [11], [14], [26], [27]. Studies have shown that the ability to express the

This research is supported in part by a grant from the AFRL Mathematical Modeling and Optimization Institute contracts #FA8651-08-D-0108/042–043 and the USDA NIFA AFRI National Robotics Initiative #2013-67021-21074. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agency. †Corresponding author: siddhart@ufl.edu.

978-1-4799-8697-2/15 $31.00 © 2015 IEEE
DOI 10.1109/SMC.2015.172

operator's intent has large preference for the interaction method [18], [28]. Taïx et al. developed a haptic arm and space mouse based user interaction tool [27] for navigating narrow spaces in higher dimensions. In [26], Ladeveze et al. proposed a visual-haptic feedback for path planning in cluttered environments.

In this paper, we extend the human-RRT collaboration results by Caves [11] and Griner [14] to path planning in urban environments. The urban environment can be characterized by cluttered configuration-space with state-space constraints (e.g., obstructions, speed limits), varying topography (e.g., traffic density, road or lane closures), high-value or high-risk zones (e.g., school areas, government buildings, military installations, nuclear facilities), unpredictable events (e.g., accidents, fire). This necessitates dynamic replanning of autonomous vehicles to accommodate the time-varying characteristics of the given environment based on new information or observations. The objective of this work is twofold. First, to develop an efficient planner for urban environments leveraging on the state-of-the-art sampling-based algorithms. Second, to develop a touch-based GUI for human collaboration following the principles of an effective GUI, i.e., learnability, controllability, and usability. In addition to defining waypoints and wayareas [11], [14] for the automated planner, the developed GUI enables the operator to specify planning space constraints, e.g., by selecting roads or by defining circular "no-grow" regions. More importantly, the presented algorithm also considers state-dependent risk to grow paths. Based on gathered intelligence and situation awareness, the operator may specify risk associated with planning paths in various regions of the configuration space. For example, in a military application of target tracking and interception, the crowded and high-value regions, such as school building and shopping malls, would have high risk to grow paths. The applications that can benefit from the developed collaborative urban path planning method include: self-driving cars, urban target tracking, urban search and rescue in disaster relief, and emergency response planning.

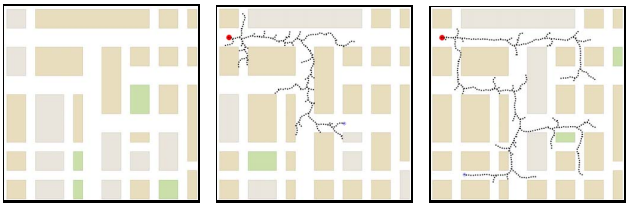
II. Path Planning Algorithm

Fig. 1(a) shows an urban environment that is cluttered with obstructions. Let $\mathbf{x}(t) \in \mathcal{X}$ be the time-varying state of an unmanned aerial vehicle (UAV), where $\mathcal{X}$ is the configuration space. The configuration space may include free space $\mathcal{X}_{free}$ and obstacles $\mathcal{X}_{obs}$, such that $\mathcal{X}_{free} \cup \mathcal{X}_{obs} = \mathcal{X}$. Given the initial configuration $\mathbf{x}_{init}$ of the UAV, the goal of path planning is to find dynamically feasible, obstacle-free paths spanning $\mathcal{X}$ (or reaching a goal configuration $\mathbf{x}_{goal}$). Beginning with $\mathbf{x}_{init}$, the standard RRT randomly samples a point $\mathbf{x}_{rand}$ from $\mathcal{X}_{free}$. Then, the tree grows from the nearest node $\mathbf{x}_{near}$ in the direction of $\mathbf{x}_{rand}$ by an amount $\varepsilon \in \mathbb{R}$ to reach the new configuration $\mathbf{x}_{new}$. Obstacle avoidance and dynamic feasibility checks may be included in this step. Along with the paths from $\mathbf{x}_{init}$, the planner also provides a series of control inputs necessary to follow the paths.

In the presented problem, for the constant altitude flight of a fixed-wing UAV, $\mathcal{X} \in \mathbb{R}^2 \times \mathbb{SO}(3)$. The constant altitude UAV dynamics are given by

$$\dot{x} = V\cos(\phi) \qquad \dot{y} = V\sin(\phi)$$
$$\dot{\phi} = \frac{g}{V}\tan(\theta) \qquad \dot{\theta} = \frac{u-\theta}{\tau} \tag{1}$$

where $(x(t), y(t)) \in \mathbb{R}^2$ is the xy-position of the UAV on the map, $\phi(t), \theta(t)$ are the heading and roll angles, respectively, and $u(t)$ is the roll rate control input. Let $\mathbf{x}(t) = [x, y, \phi, \theta]^T$ be the state vector. The forward velocity V is uncontrolled and can be state-varying, for instance, when flying in congested alleyways the velocity can be small versus when flying in open spaces. The UAV is considered to travel along the road network and does not fly over the obstacles.



(a) (b) (c)

Fig. 1. (a) An urban environment with obstructions (rectangles) and road network (empty space), (b) standard RRT path planning (runtime 100s), (c) modified RRT path planning (runtime 100s).

Typically, for an urban environment, we have $\mathcal{X}_{free} \ll \mathcal{X}_{obs}$. Therefore, the random sampling approach in RRT results in higher possibility of getting stuck in the obstacles and growing branches along the roads that go "nowhere" as shown in Fig. 1(b). We present a modified RRT algorithm with "guided" sampling for improved planning performance in congested urban environments. The presented algorithm assumes complete knowledge of the map.

Before we present the algorithm, let us define the nomenclature. The tree is an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ consisting of vertices (nodes) $\mathcal{V}$ and edges $\mathcal{E}$. The road network consists of a set of road segments R and a set of intersections I with finite area. The xy-space covered by an i^{th} intersection or road be denoted by $\mathcal{Y}(I_i) \subset \mathbb{R}^2$ or $\mathcal{Y}(R_i) \subset \mathbb{R}^2$, respectively. Let k denote the discrete time.

The main idea is, instead of uniformly sampling points from the entire free space $\mathcal{X}_{free}$, to sample more often from a feasible subset $\mathcal{X}^k_{ints} \subset \mathcal{X}_{free}$ to incrementally (and autonomously) guide the algorithm to explore the free space. The concept of "visible intersections" is introduced to obtain the desired space $\mathcal{X}^k_{ints}$. Given a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, the visible intersections are defined as a subset of I that are visible from the existing nodes $\mathcal{V}$. In other words, the nodes have an obstacle-free line-of-sight to the visible intersections. Let the set of visible intersections at time k be denoted by I^k_{vis}. Then, $\mathcal{X}^k_{ints}$ at time k is the subset of $\mathcal{X}_{free}$ with xy-space covered by $\mathcal{Y}(I^k_{vis})$, i.e., $\mathcal{Y}(I^k_{vis}) \times \mathbb{SO}(3) \to \mathcal{X}^k_{ints}$. Finally, the sampling scheme can be written as

$$\mathbb{P}(\mathbf{x}_{rand} \in \mathcal{X}^k_{ints}) = p, \ \mathbb{P}(\mathbf{x}_{rand} \in \mathcal{X}_{free}) = q$$
$$p \gg q, \ p + q = 1. \tag{2}$$

The modified-RRT pseudocode is given in Algorithm 1. 'VISIBLE_INTERSECTIONS' finds new intersections visible to $\mathcal{V}$ at time k to obtain I^k_{vis} and 'CONF_SPACE' provides the $\mathbb{R}^2$ configuration space covered by I^k_{vis}. As the tree grows, the set I^k_{vis} approaches I due to probabilistically complete nature of RRT. 'RANDOM_STATE' selects a random test point, $\mathbf{x}_{rand}$, following the sampling scheme in (2), and 'EXTEND' grows the tree based on the feasibility of $\mathbf{x}_{rand}$. Fig. 1(c) shows the path planning using the proposed modified RRT. It can be seen that the planner has a tendency to generate paths with less branches in between the intersections. The modified algorithm covers more space in the same amount of time when compared to the standard RRT (see Fig. 1(b)).

Algorithm 1 Modified RRT-based path planning

procedure BUILD_RRT
  $\mathcal{V} \leftarrow \mathbf{x}_{init}$, $\mathcal{E} \leftarrow \emptyset$, $\mathcal{G}(\mathcal{V}, \mathcal{E})$
  for $n = 1$ to N **do**
    $I^k_{vis} \leftarrow$ VISIBLE_INTERSECTIONS$(\mathcal{V})$
    $\mathcal{X}^k_{ints} \leftarrow$ CONF_SPACE(I^k_{vis})
    $\mathbf{x}_{rand} \leftarrow$ RANDOM_STATE$(\mathcal{X}_{free}, \mathcal{X}^k_{ints}, p, q)$
    $\mathcal{G}(\mathcal{V}, \mathcal{E}) \leftarrow$ EXTEND$(\mathbf{x}_{rand})$
procedure EXTEND
  $\mathbf{x}_{near} \leftarrow$ NEAREST_NEIGHBOR$(\mathbf{x}_{rand}, \mathcal{G})$
  $\mathbf{x}_{new} \leftarrow$ STEER$(\mathbf{x}_{rand}, \mathbf{x}_{near})$
  if OBSTACLE_FREE$(\mathbf{x}_{near}, \mathbf{x}_{new})$ **then**
    $\mathcal{V} \leftarrow \mathcal{V} \cup \mathbf{x}_{new}$
    $\mathcal{E} \leftarrow \mathcal{E} \cup \{\mathbf{x}_{near}, \mathbf{x}_{new}\}$

III. Human-RRT Collaboration

The objective of human-RRT collaboration is to incorporate expert knowledge into the automated path planning algorithm given in Section II for improved performance. A touch-based GUI as shown in Fig. 2 is developed. The GUI includes the map of the environment for user feedback and interaction. The functionality of the GUI is presented in the following sections.

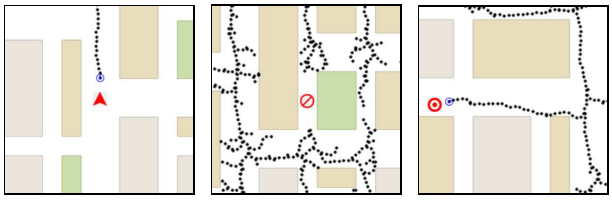
A. RRT Run Control Panel

The RRT control panel enables the operator to start and stop or pause the automated planner. It also allows the operator to control two important planning parameters, the forward velocity $V(x)$ and the tree growth factor ε, in real-time. The maneuverability of UAV depends on its forward velocity. Therefore, the automated planner chooses $V(x)$ based on the location of nodes $\mathcal{V}$; the planned path results in slower speeds along congested alleyways and intersections and higher speeds along wider roads. The growth factor ε dictates the coverage rate of the planner; higher ε covers the configuration space faster but reduces maneuverability and increases the possibility of getting stuck in the obstacles.

In certain situations, it might be desirable for the operator to override the automation-selected values of these parameters. For example, in target tracking, the operator may vary velocity according to the velocity of the target. The growth factor ε can be increased when planning in time sensitive situations that demand higher coverage. The GUI enables the operator to override V and ε by moving the slider or entering the value.

B. Gesture Control

Gesture control is designed for fast interaction to allow the operator to spend more time on high-level planning activities. It includes three commonly used motion gestures, which are wayarea, space constraints, and waypoint as shown in Fig. 3.



(a) (b) (c)

Fig. 3. User interaction using (a) wayarea, (b) space constraint, and (c) waypoint gestures.

Wayarea: In the context of urban environments, the wayarea is defined as the roads or intersections that must be included in the path by the automated planner. The gesture, when invoked, is applied with a touch or click on the map. The applied gesture is shown at the center of the wayarea for user feedback. The objective is to force the planner to generate feasible paths to go through the specified wayarea as early as possible.

Let $g_{area} \in \mathbb{R}^2$ denote the center of the wayarea. The set of intersections visible from g_{area} be I_{area}, and I_{near} be the nearest intersection, where the metric can be the Euclidean distance or the Manhattan length. Given the visible intersections I^k_{vis} to the nodes $\mathcal{V}$ at time k, the goal is to find the shortest path between I^k_{vis} and I_{near}. Let the map be modeled as an undirected weighted graph $\mathcal{G}_m(\mathcal{V}_m, \mathcal{E}_m)$, where $\mathcal{V}_m$ are the intersections, $\mathcal{E}_m$ are the connecting roads, and the edges are weighted by the length of the roads. We propose to use Dijkstra's algorithm iteratively to obtain the distance between each of the intersections in I^k_{vis} and I_{near} on $\mathcal{G}_m$. The overall shortest path I^k_{short} corresponding to the intersection with minimum distance is chosen as the desired path. To grow a branch of the tree along the desired path, we progressively direct the algorithm to visit intersections I^k_{short} by designing an appropriate sampling scheme. We know that the first intersection $I^k_{short}(1)$ is visible to the tree, i.e., $I^k_{short}(1) \in I^k_{vis}$. Therefore, the probability of sampling points with xy-coordinates from the space covered by $I^k_{short}(1)$ is held high until the tree reaches that intersection. When at $I^k_{short}(1)$, the shortest path to I_{near} is re-computed based on the revised set of visible intersections. The procedure is repeated until arriving at I_{near}. The receding horizon planning described above can take into account dynamic changes in the environment, e.g., avoid hazardous areas or track dynamic target. Once at I_{near}, the tree can be grown along the desired road or intersection by randomly sampling points from that

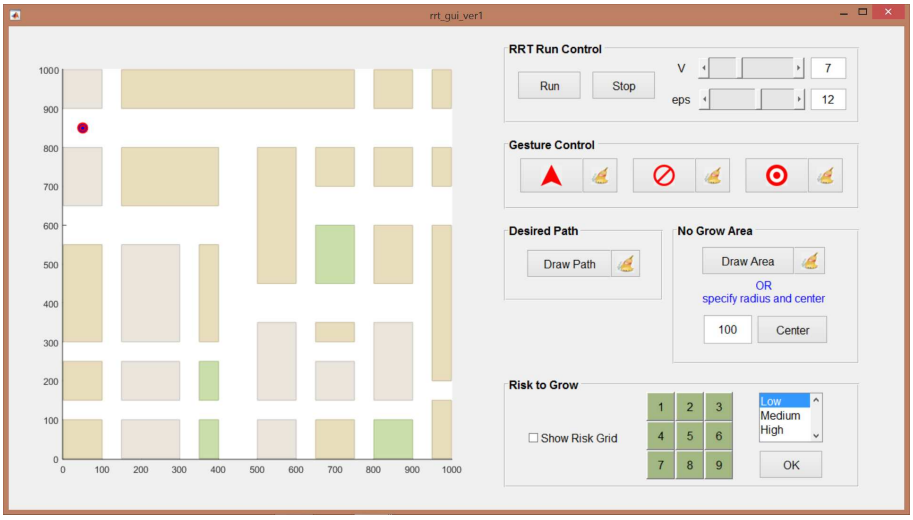


Fig. 2. Graphical user interface for human-RRT collaboration

road or intersection. The sampling pseudocode is given in Algorithm 2.

Algorithm 2 Modified RANDOM_STATE function

```
1: procedure RANDOM_STATE
2:     I^k_short ← DIJKSTRAS_SHORTEST_PATH(I^k_vis, I_near, G_m)
3:     if proj(V) ∉ Y(I^k_short(1)) then
4:         x_rand ← P(x_rand ∈ X^k_short(1)) ≫ p ≫ q
```

where $\texttt{proj}(\cdot) : \mathbb{R}^2 \times \mathbb{SO}(3) \to \mathbb{R}^2$ projects tree vertices on the xy-plane, and $\mathcal{Y}(I^k_{short}(1)) \times \mathbb{SO}(3) \to \mathcal{X}^k_{short}(1)$. Also, in Algorithm 2, $\mathbb{P}(\mathbf{x}_{rand} \in \mathcal{X}^k_{short}(1)) + p + q = 1$.

⊘ **Space Constraints:** The space constraints gesture specifies structured restrictions on the planning space, i.e., the areas to be excluded. In this work, roads or intersections are considered as structured restrictions. Similar to wayareas, the gesture uses click and drop interface where the operator selects the gesture and drops it on the desired road or intersection.

Consider the operator selects a road, R_{cons}, as a planning space constraint. Let $\mathcal{Y}(R_{cons}) \subset \mathbb{R}^2$ be the two-dimensional xy-space under R_{cons}. To account for the space constraints, the EXTEND function in Algorithm 1 is modified. In addition to obstacle check, the function checks whether the xy-position of the new point is in R_{cons} to prevent the tree from growing along the selected road (see Algorithm 3).

Algorithm 3 Modified EXTEND function

```
1: procedure EXTEND
2:     if OBSTACLE_FREE(x_near, x_new) then
3:         if proj(x_new) ∉ Y(R_cons) then
4:             V ← V ∪ x_new
5:             E ← E ∪ {x_near, x_new}
```

The other case is when the operator selects an intersection, say I_{cons}. From the graph $\mathcal{G}_m$, let the connected roads to I_{cons} be R_{cons}. Since the tree can not pass through the intersection, it may not be desirable to grow the tree along the connecting roads R_{cons}. The modified EXTEND function in Algorithm 3 can be used to avoid I_{cons} and R_{cons}. In both the cases, the weight of the edges corresponding to roads R_{cons} in $\mathcal{G}_m$ is increased to avoid planning paths through the space constraints.

⊙ **Waypoint:** Lastly, the waypoint gesture allows one to specify a point that must be included in the path. Similar to wayarea and constrained space, the waypoint gesture also relies on the click and drop interface.

Let $P_{way} \in \mathbb{R}^2$ be a waypoint selected by operator on the two-dimensional map. Also, let I_{near} be the nearest intersection to the waypoint. Algorithm 2 can be used to find the shortest path to I_{near} to grow the tree. From I_{near}, the points close to P_{way} (within a small radius w from P_{way}) are sampled to grow in the vicinity of the waypoint.

There is a provision to cancel any gesture by selecting ( ) and choosing the respective gesture on the map.

C. Desired Path

Apart from quick interaction through gestures, the operator can specify a desired trajectory for the planner to follow. For example, the operator may wish to plan a path that goes around certain building to inspect and collect information or to intercept a target as shown in Fig. 4.

The operator uses the touch screen interface to draw the desired trajectory in two-dimensional space. The goal of the RRT planner is to generate a dynamically feasible path for constant altitude flight of the UAV along the operator-defined trajectory. The trajectory can be discretized into l points $d_n \in \mathbb{R}^2$, where $n = 1, 2, \ldots, l$, and let I_{near} be the nearest intersection to d_1. The operator stroke that defines the desired trajectory has a finite area, i.e., the trajectory has a finite width. It is assumed that the width of the stroke along the trajectory is constant and denoted by $w \in \mathbb{R}$.

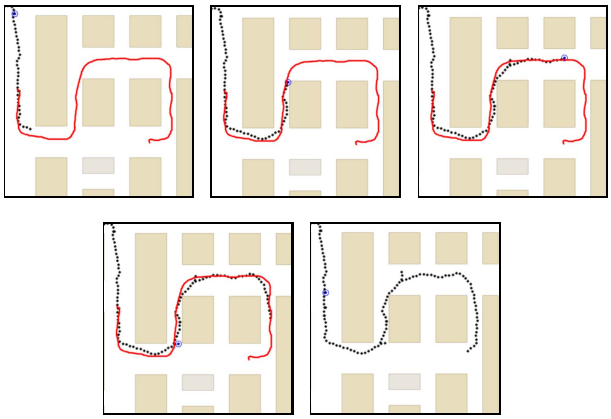


Fig. 4. Automated planner growing a feasible path along the operator-defined trajectory.

The sampling method described in Algorithm 2 can be used to grow the tree to I_{near}. The points d_n can be considered as waypoints to be followed in a defined sequence. Therefore, we progressively sample random points from a circle of radius w centered at d_n to grow nodes along the trajectory. Fig. 4 shows an example of the operator-defined trajectory (continuous red line) and the corresponding RRT-planned path (black dots).

D. No-Grow Area

The purpose of including no-grow (or do not grow) areas in addition to space constraints (Section III-B) is to provide the ability to specify more general planning restrictions. For example, to avoid collateral damage, the operator may choose not to pursue a target in populated areas.

The no-grow areas are specified as circles (see Fig. 5) and can be drawn by selecting two points on the map. The first point is the center of the circle and the distance between the two points being the radius. Alternately, the operator may select the center in the same manner, but the exact value of the radius can be entered in the text field. To maintain situation awareness, the no-grow areas are displayed with transparency.

Let A_n, $n = 1, 2, \ldots, l$, be l number of no-grow areas. The xy-space covered by A_n can be denoted by $\mathcal{Z}(A_n) \subset \mathbb{R}^2$. The edges in $\mathcal{G}_m$ corresponding to roads that are partially or completely covered by $\mathcal{Z}(A_n)$ are weighted high to avoid planning paths through the no-grow areas. The EXTEND function in Algorithm 3 is altered to include a check to find if the point $\mathbf{x}_{new}$ is in any of the no-grow areas as shown in Algorithm 4.

Algorithm 4 Modified EXTEND function

```
1: procedure EXTEND
2:     if OBSTACLE_FREE(x_near, x_new) then
3:         if proj(x_new) ∉ 𝒴(R_cons) then
4:             if proj(x_new) ∉ 𝒵(A_n) then
5:                 𝒱 ← 𝒱 ∪ x_new
6:                 ℰ ← ℰ ∪ {x_near, x_new}
```

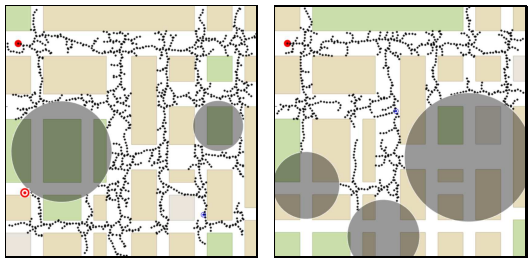
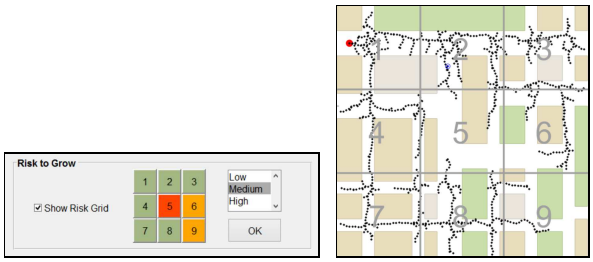


Fig. 5. Automated planner avoiding no-grow areas (gray circles).



(a) (b)

Fig. 6. (a) Interface for defining risk, (b) Feasible paths generated by an automated planner taking into account the state-dependent risk.

E. Risk to Grow

The gathered situation awareness can be used to effectively manage the planner. The state-dependent risk or penalty is one approach to encode the situation awareness. Consider the following examples for clarity. When the UAV is operating in an hostile environment, the operator may specify the risk of the UAV getting shot down for different regions such that the high-risk areas should be avoided as far as possible. In another example of urban target tracking, the operator may have a good estimate of the target location based on gathered evidence. The location estimate can be encoded in the form of risk to efficiently search the target. In this context, the risk is the likelihood of finding the target and hence, high-risk areas area favorable.

In the presented example, we consider the first notion of risk, i.e., penalty to grow the tree. The map is divided in nine equal area cells. The risk value associated with each cell can be low, medium, or high. The operator selects the region using the numerical pad with $1-9$ keys, selects the risk value and presses 'OK'. The operator can view the nine cells on the map using the 'Show Risk Grid' option. Fig. 6(a) shows an example where the operator has selected medium risk for regions 6 and 9, and high risk for 5. The default risk value is low.

Let $\mathcal{Z}(C_n^v) \in \mathbb{R}^2$ be the xy-space covered by the C_n^v, $n = 1, 2, \ldots, 9$ cells and $v = \{lo,med,hi\}$ be the risk value – low, medium, and high, respectively. The growth of the RRT is biased based on the risk value associated with each cell. The probability of sampling from or growing in a high risk cell is small compared to that of the medium and low risk cells. In Algorithm 5, the probability p^v is such that $p^{hi} < p^{med} < p^{lo}$. The performance of risk-based planning for the example given

in Fig. 6(a) is shown in Fig. 6(b).

Algorithm 5 Modified EXTEND function

```
1: procedure EXTEND
2:     if OBSTACLE_FREE(x_near, x_new) then
3:         if proj(x_new) ∉ 𝒴(R_cons) then
4:             if proj(x_new) ∉ 𝒵(A_n) then
5:                 if proj(x_new) ∉ 𝒵(C_n^v) then
6:                     ℙ(𝒱 ← 𝒱 ∪ x_new) < p^v
7:                     ℙ(ℰ ← ℰ ∪ {x_near, x_new}) < p^v
```

IV. Conclusion

Human-RRT collaborative algorithm and graphical user interface is developed for path planning in urban environments. A modified-RRT algorithm is presented to improve automated planning. The user interface consists of the map of the environment and a set of tools for interacting with the planner. Simple gesture controls are included to specify wayareas, space constraints, and waypoints. The user can also draw a desired trajectory for the planner to follow using touch interface. In addition, to feed the operator's situation awareness into the planner, the operator can specify the state-dependent risk using the interface.

The future work will consider evaluating and improving the user interface using experiments involving human subjects. Another avenue for future research is to estimate human intentions to autonomously affect planning, build trust, and reduce human interaction.

References

[1] T. Hoeniger, "Dynamically shared control in human-robot teams through physical interactions," in *Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on*, vol. 2. IEEE, 1998, pp. 744–749.

[2] S. S. Mehta, M. McCourt, E. A. Doucette, and J. W. Curtis, "A touch interface for soft data modeling in bayesian estimation," in *Proc. of the IEEE Conference on Systems, Man, and Cybernetics (SMC)*, San Diego, CA, 2014, pp. 3732–3737.

[3] A. Dani, M. McCourt, J. W. Curtis, and S. S. Mehta, "Information fusion in human-robot collaboration using neural network representation," in *Proc. of the IEEE Conference on Systems, Man, and Cybernetics (SMC)*, San Diego, CA, 2014, pp. 2114–2120.

[4] T. B. Sheridan, *Humans and automation: System design and research issues*. John Wiley & Sons, Inc., 2002.

[5] T. Fong, C. Thorpe, and C. Baur, "Robot, asker of questions," *Robotics and Autonomous systems*, vol. 42, no. 3, pp. 235–243, 2003.

[6] M. L. Cummings, S. Bruni, S. Mercier, and P. Mitchell, "Automation architecture for single operator, multiple UAV command and control," Massachusetts Inst. of Tech., Cambridge, Tech. Rep., 2007.

[7] S. S. Mehta, P. E. K. Berg-Yuen, E. L. Pasiliao, and R. A. Murphey, "A control architecture for human-machine interaction in the presence of unreliable automation and operator cognitive limitations," in *Proc. of AIAA Guidance, Navigation and Control (GNC) Conf.*, Minneapolis, MN, 2012, pp. AIAA 2012–4543.

[8] M. L. Cummings, C. E. Nehme, J. Crandall, and P. Mitchell, "Predicting operator capacity for supervisory control of multiple UAVs," in *Innovations in Intelligent Machines-1*. Springer, 2007, pp. 11–37.

[9] A. Kott, R. Budd, L. Ground, L. Rebbapragada, and J. Langston, "Building a tool for battle planning: challenges, tradeoffs, and experimental findings," *Applied Intelligence*, vol. 23, no. 3, pp. 165–189, 2005.

[10] M. P. Linegang, H. A. Stoner, M. J. Patterson, B. D. Seppelt, J. D. Hoffman, Z. B. Crittendon, and J. D. Lee, "Human-automation collaboration in dynamic mission planning: A challenge requiring an ecological approach," in *Proceedings of the Human Factors and Ergonomics Society 50th Annual Meeting*, 2006, pp. 2482–2486.

[11] A. D. J. Caves, "Human-automation collaborative RRT for UAV mission path planning," Master's thesis, Massachusetts Institute of Technology, 2010.

[12] I. Maza, K. Kondak, M. Bernard, and A. Ollero, "Multi-UAV cooperation and control for load transportation and deployment," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, pp. 417–449, 2010.

[13] M. L. Cummings, J. Marquez, and N. Roy, "Human-automated path planning optimization and decision support," *International Journal of Human-Computer Studies*, vol. 70, no. 2, pp. 116–128, 2012.

[14] A. Griner, "Human-RRT collaboration in unmanned aerial vehicle mission path planning," Master's thesis, Massachusetts Institute of Technology, 2012.

[15] A. Kopeikin, A. Clare, O. Toupet, J. P. How, and M. L. Cummings, "Flight testing a heterogeneous multi-UAV system with human supervision," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2012.

[16] M. Lewis, "Human interaction with multiple remote robots," *Reviews of Human Factors and Ergonomics*, vol. 9, no. 1, pp. 131–174, 2013.

[17] J. Y. Chen, "UAV-guided navigation for ground robot tele-operation in a military reconnaissance environment," *Ergonomics*, vol. 53, no. 8, pp. 940–950, 2010.

[18] M. L. Cummings, J. P. How, A. Whitten, and O. Toupet, "The impact of human–automation collaboration in decentralized multiple unmanned vehicle control," *Proceedings of the IEEE*, vol. 100, no. 3, pp. 660–671, 2012.

[19] S.-Y. Chien, H. Wang, and M. Lewis, "Human vs. algorithmic path planning for search and rescue by robot teams," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 54, no. 4. Sage Publications, 2010, pp. 379–383.

[20] J. Y. Chen and M. J. Barnes, "Supervisory control of multiple robots effects of imperfect automation and individual differences," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 54, no. 2, pp. 157–174, 2012.

[21] M. Lewis, J. Polvichai, K. Sycara, and P. Scerri, "17. scaling-up human control for large UAV teams," *Human factors of remotely operated vehicles*, vol. 7, pp. 237–250, 2006.

[22] A. S. Clare, M. L. Cummings, J. P. How, A. K. Whitten, and O. Toupet, "Operator object function guidance for a real-time unmanned vehicle scheduling algorithm," *Journal of Aerospace Computing, Information, and Communication*, vol. 9, no. 4, pp. 161–173, 2012.

[23] A. S. Clare, P. C. Maere, and M. L. Cummings, "Assessing operator strategies for real-time replanning of multiple unmanned vehicles," *Intelligent Decision Technologies*, vol. 6, no. 3, pp. 221–231, 2012.

[24] X. Sun, C. Cai, and X. Shen, "A new cloud model based human-machine cooperative path planning method," *Journal of Intelligent & Robotic Systems*, pp. 1–17, 2014.

[25] S. M. LaValle, "Rapidly-exploring random trees a new tool for path planning," Computer Science Dept., Iowa State University, Tech. Rep. 98-11, Oct. 1998.

[26] N. Ladeveze and J.-Y. Fourquet, "On the collaboration of an automatic path-planner and a human user for path-finding in virtual industrial scenes," in *Control Automation Robotics & Vision (ICARCV), 2010 11th International Conference on*. IEEE, 2010, pp. 467–472.

[27] M. Taïx, D. Flavigné, and E. Ferré, "Human interaction with motion planning algorithm," *Journal of Intelligent & Robotic Systems*, vol. 67, no. 3-4, pp. 285–306, 2012.

[28] E. M. Roth, M. L. Hanson, C. Hopkins, V. Mancuso, and G. L. Zacharias, "Human in the loop evaluation of a mixed-initiative system for planning and control of multiple uav teams," in *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 48, no. 3. SAGE Publications, 2004, pp. 280–284.