improving path planning of unmanned aerial vehicles in an immersive environment using meta-paths and terrain information

Levi Daniel Swartzentruber
Iowa State University

Improving path planning of unmanned aerial vehicles in an immersive environment using meta-paths and terrain information

by

Levi Daniel Swartzentruber

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Co-majors: Mechanical Engineering; Human Computer Interaction

Iowa State University

Ames, Iowa

2009

Table of Contents

List of Figures

List of Tables

Abstract

Effective command and control of unmanned aerial vehicles (UAVs) is an issue under investigation as the military pushes toward more automation and incorporation of technology into their operational strategy. UAVs require the intelligence to maneuver safely along a path to an intended target while avoiding obstacles such as other aircraft or enemy threats. To date, path-planning algorithms (designed to aid the operator in the control of semi-autonomous UAVs) have been limited to providing only a single solution (alternate path) without utilizing input or feedback from the UAV operator. The work presented in this thesis builds off of and improves an existing path planner. The original path planner presents a unique platform for decision making in a three-dimensional environment where multiple solution paths are generated using Particle Swarm Optimization (PSO) and returned to the operator for evaluation. The paths are optimized to minimize risk due to enemy threats, to minimize fuel consumption incurred by deviating from the original path, and to maximize reconnaissance over predefined targets. The work presented in this thesis focuses on improving the mathematical models of these objectives. Terrain data is also incorporated into the path planner to ensure that the generated alternate paths are feasible and at a safe height above ground.

An effective interface is needed to evaluate the alternate paths returned by PSO. A “meta-path” is a new concept presented in this thesis to address this issue. Meta-paths allow an operator to explore paths in an efficient and organized manner by displaying multiple alternate paths as a single path cloud. The interface was augmented with more detailed information on these paths to allow the operator to make a more informed decision. Two other interaction techniques were investigated to allow the operator more interactive control over the results displayed by the path planner. Viewing the paths in an immersive environment enhances the operator's understanding of the situation and the options while facilitating better decision making. The problem formulation and solution implementation are described along with the results from several simulated scenarios. Preliminary assessments using simulated scenarios show the usefulness of these features in improving command and control of UAVs.

Finally, a user study was conducted to gauge how different visualization capabilities affect operator performance when using an interactive path planning tool. The study demonstrates that viewing alternate paths in 3D instead of 2D takes more time because the operator switches between multiple views of the paths but also suggests that 3D is better for allowing the operator to understand more complex situations.

Chapter 1 – Introduction

Military usage of unmanned systems

Military operations increasingly incorporate unmanned systems at sea, on the ground, and in the air. The mandate for utilizing these advanced vehicles in combat situations comes from the top levels of government [1]. Motivation for doing so springs from the ability such vehicles have for reducing military costs and preserving the lives of service personnel. The definition of unmanned systems has expanded to include aerial vehicles, ground vehicles, water surface vehicles, and submersibles.

Unmanned Aerial Vehicles (UAVs) form a subset of the unmanned systems. Figure 1 shows the Predator [2] and the X-45 [3], two of the larger UAVs that see frequent use in military operations. In Operation Enduring Freedom and Operation Iraqi Freedom, almost 400,000 flight hours have been logged by UAVs, not including hand-launched systems [4]. The combat commanders recognize the important role unmanned vehicles fill in military situations (such as gathering reconnaissance data or disposing of improvised explosive devices) and are pushing for further development and incorporation of these technologies. The Department of Defense Unmanned Systems Roadmap 2007-2032 details a vision for the development of all types of unmanned systems [4]. It is their goal to seamlessly integrate manned and unmanned systems for optimal mission execution. According to the timeline, UAVs will be able to perform a full range of mission tasks by 2030 including: surveillance, counter air strikes, penetrating strikes, and airlifts. There will be a need for increased autonomy as these vehicles perform more complex tasks in dynamic environments.

Automation will by no means eliminate the need for human involvement in the operation of the unmanned systems but does alter what this interaction looks like. Operators will move into supervisory roles, monitoring and guiding the activities of the vehicle through computer interfaces rather than mechanically controlling its movements.

Figure 1. Pictures of the Predator (left) and X-45 (right) UAVs

UAV command and control

Unmanned systems would span vehicles that are fully operator-controlled through ones that are completely autonomous. Currently, all UAVs are at least partially controlled from ground control stations. The control station for the Predator is pictured in Figure 2 and is typical for large unmanned aerial vehicles [5]. Two trained operators are needed to control one UAV at all times. One person flies the UAV. This individual is required to have a full flight training background. The tools currently available to this pilot are: statistics for the vehicle (speed and altitude), a 2D map of the area of operation with information such as known enemy locations, and a soda straw view of the world from the perspective of that UAV. The other manipulates the sensors and monitors their output. These responsibilities are taxing enough to require the operators to switch off every few hours while conducting missions. This method of operation, while functional, leaves much room for improvement as it limits the operator's situational awareness and increases the difficulty of their task.

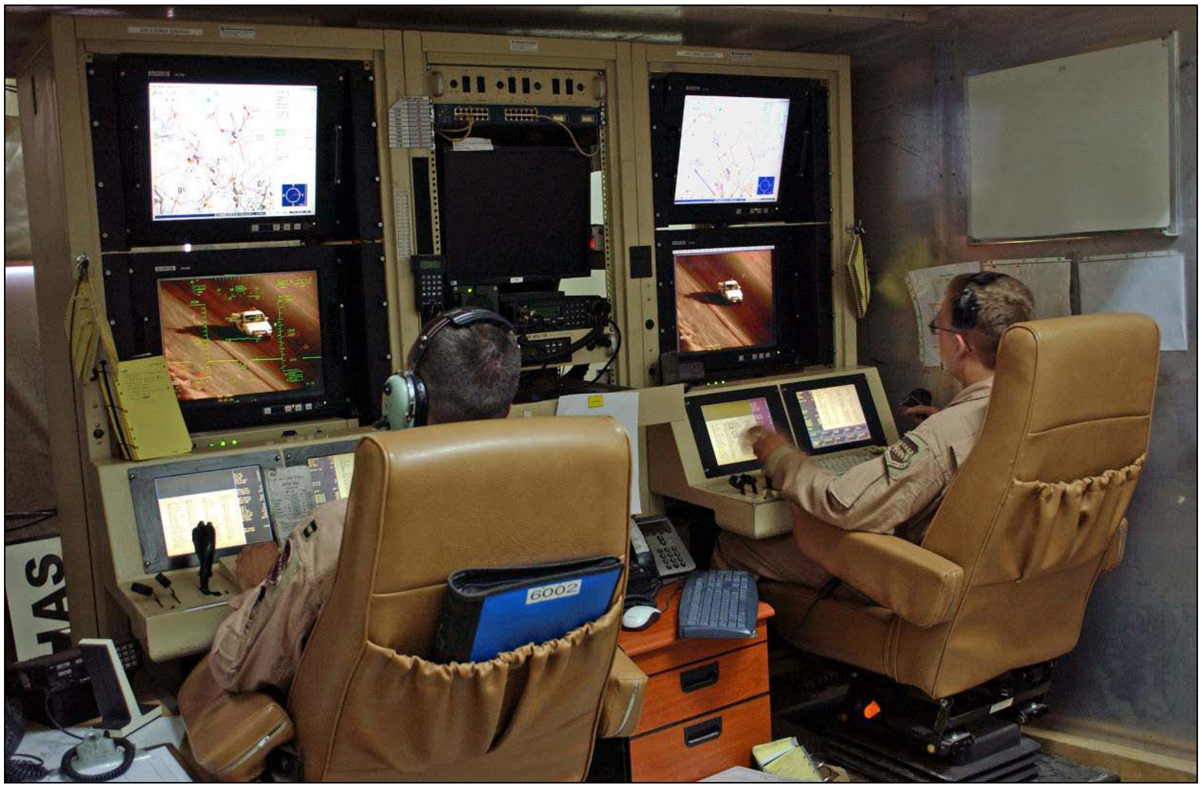


Figure 2. Predator ground control station

Situational awareness is a measure of how well the UAV operator is able to understand what the UAV is doing and what is going on in the environment around the vehicle. The soda straw view of the environment provided by UAV sensors limits what the operator sees and hinders the person's ability to develop good situational awareness. Especially in situations where the operator would be asked to control multiple vehicles, this interface is inadequate for effective command and control. The operator needs a comprehensive view of the environment, incorporating the information provided by all the collaborating vehicles, instead of disjoint data coming from each UAV separately to be able to task them jointly in an efficient manner. Then the operator would control the UAVs in this virtual, collaborative environment and have the real vehicle respond accordingly; this is a complete reversal of the current interaction technique.

The current control setup does not scale well to combat scenarios where unmanned systems comprise a significant percentage of the total number of vehicles involved. One problem would be the lack of trained personnel to function as operators. This is largely a human-computer interaction problem. The operator interface needs to allow for effective control of increasingly complex vehicles. Having the same number of individuals controlling the same number of UAVs will not solve the operator bottleneck. Groups such as the Air Force Research Laboratory are looking beyond interfaces that simply keep pace with UAV technology to ones that allow a single operator to control multiple vehicles [6]. This is a challenge that involves developing new control and display technologies, incorporating decision support aids, and building a system architecture that supports multiple UAVs.

While the interface is changing, the scope of what the UAV will be allowed to do autonomously or semi-autonomously will also need to change. If the operator is required to operate multiple UAVs, the individual’s attention will be divided between the vehicles. For this type of system to be effective, UAVs need to have autonomous operation as the default with operator intervention being the exception. The less time the operator must spend focused on a single vehicle, the more vehicles that operator can simultaneously control or the more time the operator has to maintain an overall awareness of the situation. In this way, the operator role transitions from that of a remote pilot to that of a supervisor.

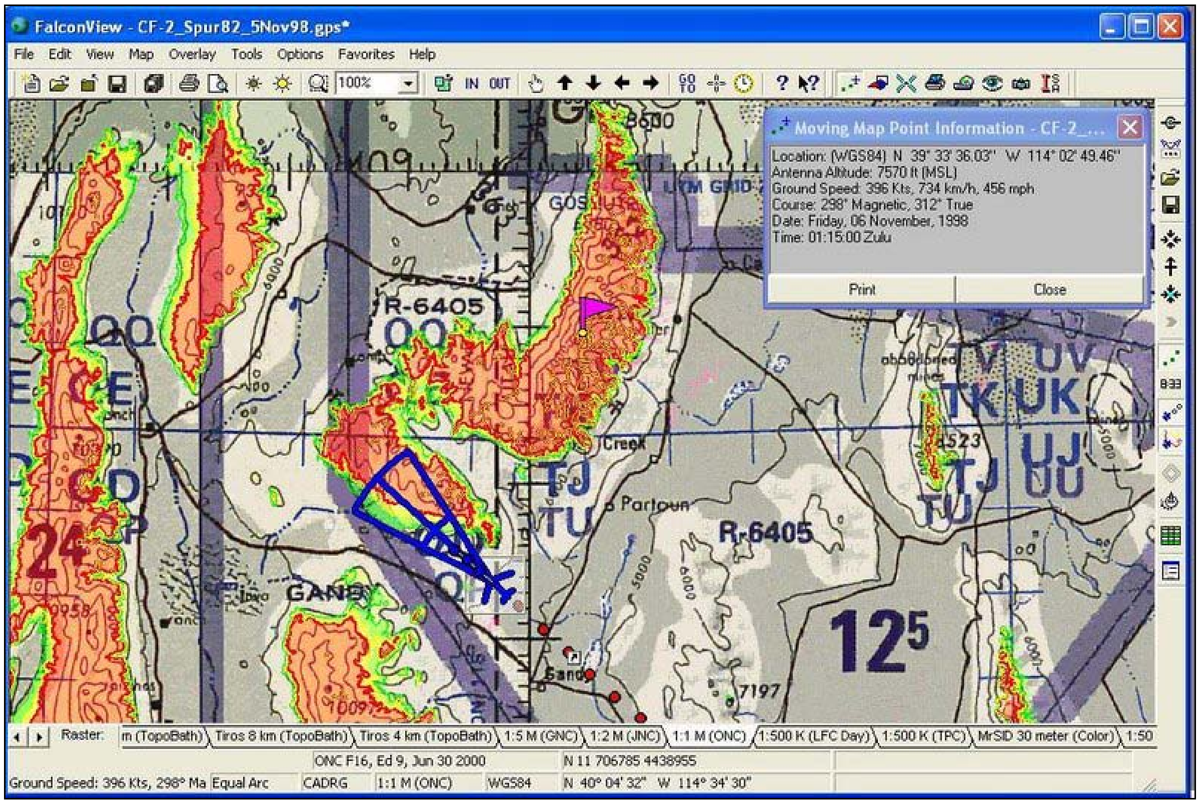
The supervisory role necessitates higher-level interaction between the individual and the vehicle. Information will need to be abstracted for the operator to avoid information overload. Currently, UAVs supply enough data to occupy the cognitive abilities of two trained operators. This information will have to be abstracted in some manner to reduce the cognitive load of each vehicle to allow one individual to control more than one UAV at a time. Just as the operator's interaction and control move to higher levels, the understanding of the details of a UAV's situation also need to move to higher levels. As the operator's time is divided between multiple UAVs, less time will be spent on any given vehicle and therefore that vehicle will have to be trusted to do more autonomously [7].

With UAVs operating largely autonomously and the operator simply supervising the vehicles, it would fall to the vehicle to recognize situations in which the operator would need to take action. Ideally, the information presentation would be such that the operator would be able to anticipate when the UAV might encounter problems and know when to intervene, but this is not likely to be the case all the time. This leads to the question: When should a UAV alert the operator that human intervention is needed and what should that intervention look like?

Interface design becomes critical as the interface is the means by which the operator gathers information about a vehicle and sends commands back to it. As the UAV is in flight, the operator must deal with situations that arise in real time. This further constrains the interface design to present information to the operator that can be quickly absorbed and acted upon. Details presented in a clear and concise manner quicken the operator's understanding and allow the individual more time to react, reducing misunderstandings and perceived workload. This can be the difference between a costly mistake and a successful mission.

The research challenge: path-planning for UAVs

One situation that might arise for an autonomously operating UAV is the need to re-plan part of its path. Path-planning is the process of automatically generating an alternate path for an unmanned vehicle [9]. In-flight, or on-line, path-planning is a complex problem to solve. Predefined criteria are needed to select one path from among the possibilities. What the criteria, or objectives, are and their relative importance determines the path or paths that will be generated as alternatives to the original. It is rarely possible to generate alternative paths that meet all the desired objectives exactly. For example, a UAV may be sent out on a reconnaissance mission with a few targets to observe, such as three bridges over a river in enemy territory. Before the UAV takes off, a flight plan is generated that will allow it to pass over all the target locations. As the UAV flies along, it may encounter an unexpected threat, such as a surface-to-air missile (SAM) site, to which it will pass dangerously close in the immediate future if its course is unaltered. Avoiding the threat will require the UAV to bypass the optimal viewing location for the target. FalconView is a software package widely used by the Department of Defense to monitor and control UAVs in this type of situation [8]. A screenshot is shown in Figure 3 where the blue glyph represents the airplane.


FalconView - CF-2_Spur82_5Nov98.gps*
File Edit View Map Overlay Tools Options Favorites Help
100%
IN OUT
Moving Map Point Information - CF-2_...
Print
Close
Raster: m (TopoBath) Tiros 8 km (TopoBath) Tiros 4 km (TopoBath) 1:5 M (GNC) 1:2 M (JNC) 1:1 M (ONC) 1:500 K (LFC Day) 1:500 K (TPC) MrSID 30 meter (Color) 1:50
R-6405
GANDY
Partoun
Creek
UTM GRID
7197

Location: (WGS84) N 39° 33' 36.03" W 114° 02' 49.46"
Antenna Altitude: 7570 ft (MSL)
Ground Speed: 396 Kts, 734 km/h, 456 mph
Course: 298° Magnetic, 312° True
Date: Friday, 06 November, 1998
Time: 01:15:00 Zulu

ONC F16, Ed 9, Jun 30 2000 | N 11 706785 4438955
Ground Speed: 396 Kts, 298° Ma | Equal Arc | CADRG | 1:1 M (ONC) | WGS84 | N 40° 04' 32" W 114° 34' 30"

Figure 3. FalconView interface for controlling UAVs

The problem is additionally complicated by the addition of a time constraint. The UAV will come within range of the threat at a point in time soon after it is discovered, perhaps 30 seconds or less. Any path-planning or alternate path selection should be completed and implemented before that time to be an effective tool for UAV control.

Optimization

While finding a path that meets the objectives at least partially is good, it would be ideal to find an optimal path for the UAV. Optimization, generally, is the process of finding the best solution to a problem, however that problem may be defined [10]. Almost all optimization of consequence is performed using numerical methods on a computer.

Numerical optimization is a systematic process of moving through or exploring the design space to find the best possible solution. A design space is the set of all solutions to the problem. In the case of path planning for UAVs, it is the set of all paths the vehicle could take. To choose one path, there needs to be a way to compare the paths. The comparison is made on the basis of a mathematical function (called the cost function) which is evaluated for each path. The objective function is developed by combining mathematical models of the user-defined criteria or objectives for that particular problem. When the optimization is complete, it should produce a solution that best matches and balances the desires of the operator as defined in the objective function—if such a solution exists. There are a wide range of techniques, algorithms, and methods for performing optimization. These methods are categorized based on the type of problem they can handle.

Heuristic methods search the design space using a population of potential solutions in an attempt to arrive at an optimal solution [11]. Genetic algorithms [12], simulated annealing [13], and particle swarm optimization [14] are all very good at dealing with multi-modal design spaces and high dimension problems.

Path-planning lends itself well to optimization with heuristic methods. It is a multi-dimensional problem as one solution is a set of points in three-dimensional space that define a path. It is also assumed to be multi-modal because there may be several possible paths that are equally optimal. Figure 4 illustrates why this might happen in a 3D search. A path that goes over a threat, the middle of the three green choices, may be equally preferable to paths that go around the threat. This creates a difficult decision for a human operator who must choose between several good possibilities. The heuristic optimization algorithms mentioned are designed to handle multi-dimensional, multi-modal problems of this nature and find a good solution, aiding the operator in solving this complex problem. Previous work on the path planner was designed to meet these challenges [9].

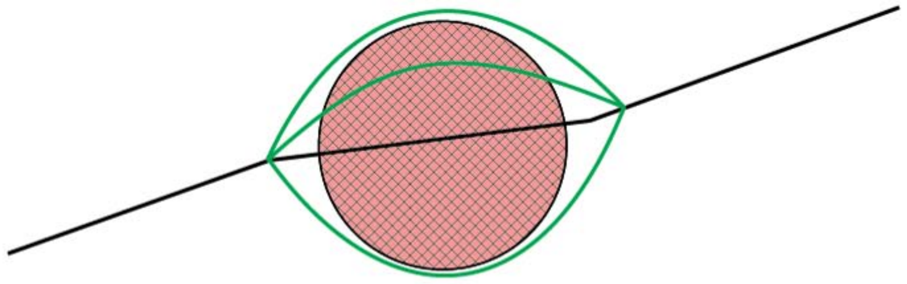


Figure 4. Path through cross-hatched hemispherical threat zone with three equally good alternatives

Statistical analysis has shown that while the heuristic methods mentioned return high-quality solutions to a wide variety of multi-modal unconstrained problems, PSO was more computationally efficient [17]. Therefore PSO is being used in this research. Particle Swarm Optimization was used to generate three alternate paths (one weighted for fuel efficiency, one for reconnaissance and one for threat avoidance). The operator could then evaluate these optimal paths and choose one for re-tasking the UAV. Equations 1, 2, and 3 outline the core of the PSO algorithm.

$$V_{iter+1} = w_{iter} + c_1 rand_p(\ )\left[pBest_i(\ ) - X_i(\ )\right] + c_2 rand_g(\ )\left[gBest_i(\ ) - X_i(\ )\right] \tag{1}$$

$$w_{iter+1} = w_{iter}\lambda_w \tag{2}$$

$$X_{iter+1} = X_{iter} + V_{iter+1} \tag{3}$$

The first equation is the velocity vector update equation. The best point found by an individual particle is its *pBest* and the best point found among all the swarm members is *gBest*. Equation 2 shows how the inertia weighting factor is adjusted. Equation 3 is the position update equation which combines the new velocity with the previous location to get the new location of the swarm member.

Visualization

Once a solution is generated to an optimization problem, an operator will need to evaluate it. The presentation of the solution is often given far less attention than the methodology in creating it, and this can lead to output that is difficult or time-intensive to interpret. In path-planning situations wherein the operator has little time to respond to the situation and still has other UAVs to monitor, it is even more important to consider carefully information presentation.

Path planners often represent the course of a UAV in 2D or 3D graphical form. The 2D case is far simpler to generate, but the elevation information must be presented to the user in some other manner. A 3D view of the path is far more intuitive as it presents the data unaltered to the operator, but it is more difficult to create and it can be difficult to determine the 3D trajectory of a path on a 2D screen. The 3D representation could have greater effectiveness if incorporated into a 3D immersive environment. From its original formulation, the path planner was designed to operate in an immersive 3D environment to most accurately present the alternate paths to the operator.

Another aspect of the visualization is the amount of additional data the operator is given as well as the means by which the operator interacts with that data and with the path itself. Simply having the path presented does not give the operator an answer to the question of how optimal the path is or how the cost of the alternate path compares with that of the original path. If there are multiple objectives included in the optimization problem, the operator may find it useful to know how a path ranks in each category as well as overall. And if the operator is given multiple paths from which to choose, then there needs to be an effective method for path comparison and selection. Each of these details should be considered when developing an interface to use with a path planner.

Motivation

Path-planning for UAVs is a complex problem for which a comprehensive solution has yet to be offered, though much progress has been made. Input for that solution is needed from several areas. Two of these areas have been addressed by the previous work on this path planner [9]. First, there was a need to create the solution within an architecture that promotes effective control and supervision of the UAV by the operator. Second, some form of optimization in the path generation process was needed to aid the operator in the decision making process by presenting that person with a few good solutions from which to choose.

Two other areas were touched on by previous work and are addressed in more detail in this thesis. The first is the challenge of formulating the problem with as complete a representation of the situation and the capabilities of the vehicle as possible. The second is carefully designing the interface as that is the operator’s window to the UAV and should facilitate informed tasking of the UAV.

This thesis presents work done to enhance and improve the path planner. Attention was given to reformulate the objectives by addressing the issue of terrain avoidance and reformulating the reconnaissance cost. The second area of research was in presenting the operator with even more choices and tools to evaluate those alternate paths to improve decision making in path planning situations for enhanced command and control.

In chapter two, a detailed literature review of current work will be discussed. Chapter three presents the method development of the path planning algorithm, interface, and interaction techniques incorporated in this path planner. Chapter four discusses the test cases performed. In chapter five, a user study is presented that analyzes the effectiveness of the interface in aiding operators in selecting an optimal path. Chapter six outlines conclusions and future work.

Chapter 2 – Background

As seen in the introduction, multiple fields must be synthesized to produce a comprehensive path planner for UAVs. Visualization, optimization, path planning, and decision making are a few of the focus areas that have valuable contributions to make. Substantial progress in path planning has already been seen as many different individuals and groups have been working on the path planning problem concerning autonomous and semi-autonomous vehicles. This wide base of research, touching on each of the areas mentioned, informs the creation of the path planner discussed in this thesis.

Virtual Battlespace

This path planner was built as a component of the Virtual Battlespace environment [18]. Development of the Virtual Battlespace commenced in 2000 when a research team at Iowa State University's Virtual Reality Applications Center (VRAC) began collaborating with the Air Force Research Lab's Human Effectiveness Directorate and the Iowa National Guard's 133rd Air Control Squadron. The goal of this project was to create an immersive virtual reality (VR) system for distributed mission training. Figure 5 shows the Virtual Battlespace running in the C6 immersive virtual reality environment. Virtual Battlespace integrates information about tracks, targets, sensors and threats into an interactive virtual reality environment. It fuses the available information about the battlespace into a coherent picture that can be viewed from multiple perspectives and scales [15, 16]. The environment and all the features are controlled using a wireless gamepad.

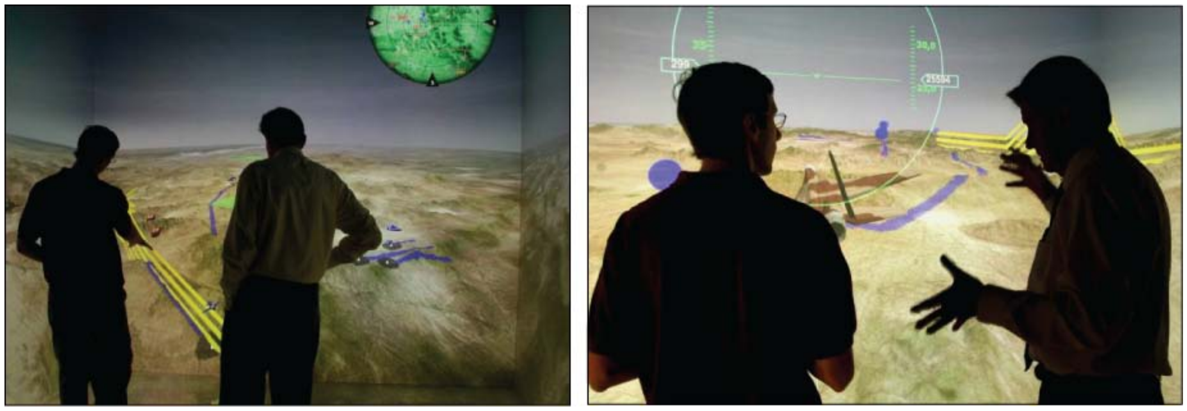
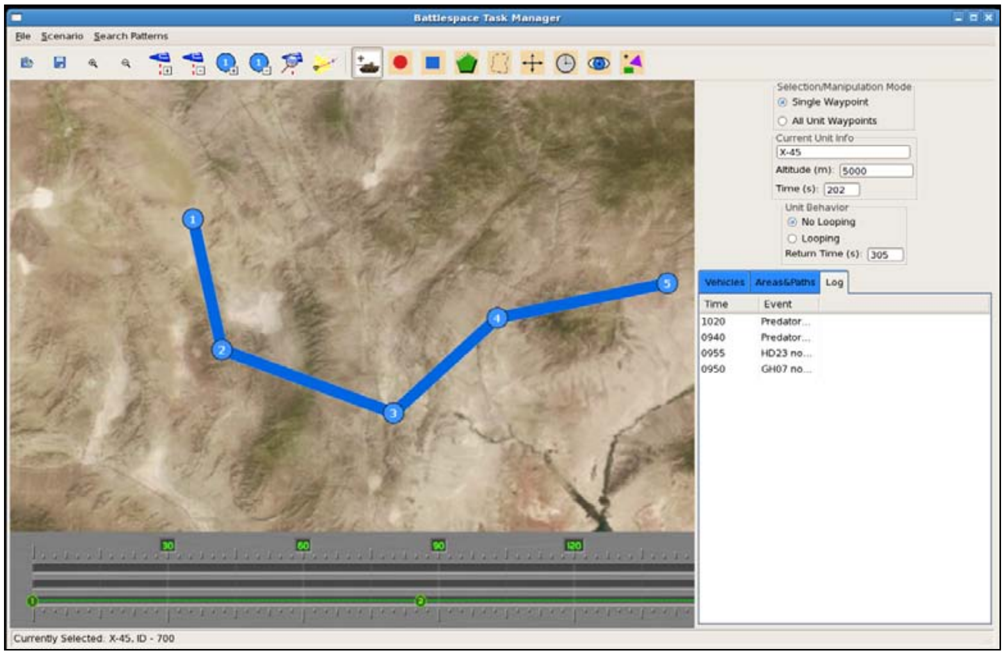


Figure 5. Virtual Battlespace in the C6 at Iowa State University

From its inception, the path planner was designed to operate in the immersive, 3D environment of the Virtual Battlespace and to utilize fully the capabilities of the C6. A path is defined by a series of points, each with a specified location in space (x, y, z position) and time (when the entity will reach that point). The scenario generator is used to specify the waypoints and so establish the initial path for the UAV to follow. This interface is shown in Figure 6 with a UAV path in blue. In Virtual Battlespace, the path is then drawn as a yellow fence with each horizontal bar representing 1000 feet of altitude. The waypoints are bends in the yellow path and the orange rectangles in the bottom picture are gates the UAV flies through along that path. Two views of the fence, with the UAV flying along it, are presented in Figure 7. With 3D graphics capability, no artificial restrictions are placed on the path display as there is one-to-one mapping from path point coordinates to coordinates in the virtual world.



Selection/Manipulation Mode
[✓ Single Waypoint] All Unit Waypoints

Current Unit Info
X-45
Altitude (m): 5000
Time (s): 202

Unit Behavior
[✓ No Looping] Looping
Return Time (s): 305

Vehicles | Areas&Paths | Log

Time	Event
1020	Predator...
0940	Predator...
0955	HD23 no...
0950	GH07 no...

Currently Selected: X-45, ID - 700

Figure 6. Scenario generator for creating UAV flight paths for the Virtual Battlespace

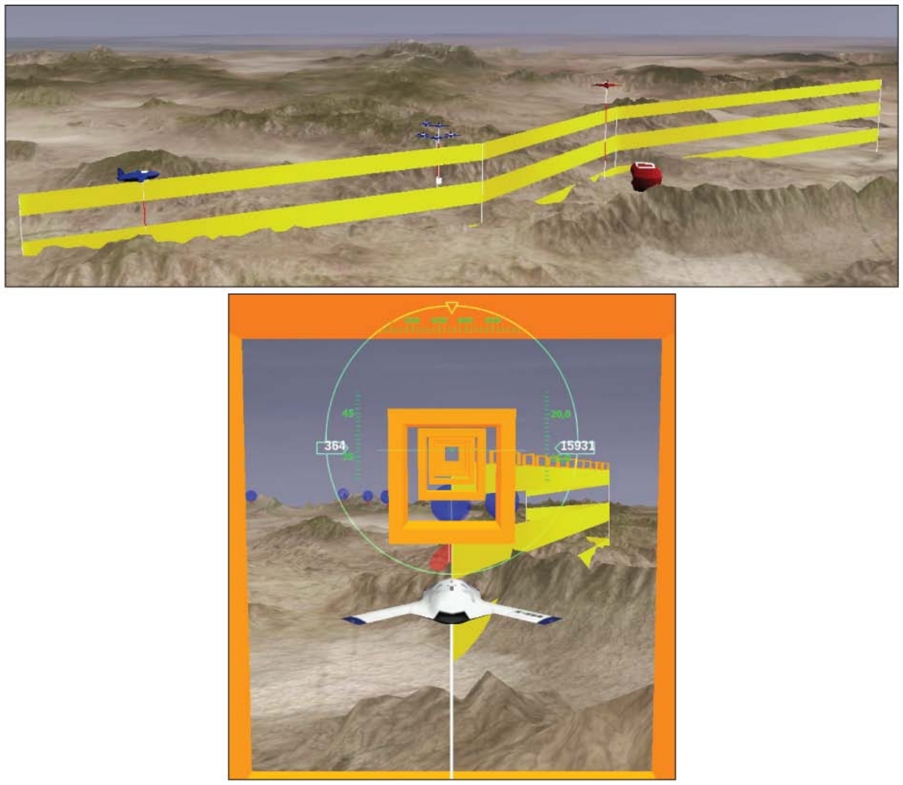


Figure 7. UAV path shown as a yellow fence (top) and a close-up of UAV flying along path (bottom) in the Virtual Battlespace

The current method of redirecting UAVs is to manually plot a new course. While manual input assures that at least one path is generated and implemented in a timely manner, this method has its drawbacks. The quality of the solution is highly dependent on the skill of the planner. For given situation, there are more alternate path choices than a human operator would be able to analyze. The operator's judgment is an important part of path planning but it would be very valuable to have a tool (a path planner) to assist the operator in this process.

Path Planning

Path planning has been and continues to be an extensively researched topic. The problem has been addressed in situations ranging from controlling the motion of computer-generated entities through virtual worlds in video games to the autonomous motion of a rover through a physical building with no prior knowledge of the environment. Much research has focused on path planning for UAVs as it is a relatively complex problem. A representative sample of these path planners will be discussed.

Many UAV path planners simplify the problem by limiting the vehicle to maneuver only in a horizontal plane and treating altitude as a constant [19]. Formulating the problem with only four degrees of freedom facilitates the application of ground vehicle path planning algorithms directly to UAVs. Two-dimensional path planners also lend themselves well to visualization on 2D computer screens. Reducing the problem from six degrees of freedom to four also exponentially reduces the computational complexity of the problem. Simpler computations are attractive because they are easier to program, and take less time to execute. In the case of

UAVs, the benefits of simplicity do not justify the cost of neglecting the 3D maneuvering capabilities of these vehicles.

Recently, many researchers have realized the limitations of two-dimensional UAV path planning and begun implementing three-dimensional path planners, taking advantage of the additional two degrees of freedom, when generating alternate paths [20, 21]. With that freedom come new challenges in path computation and visualization. To date, no research group has produced a path planner that satisfactorily addresses all of the challenges presented by this problem. Instead, groups of similar algorithms have emerged that address one aspect of the problem well while neglecting other areas.

Offline path planners tend to be the most sophisticated or complete branch of solutions to the problem. They are designed to generate the entire path of a UAV from takeoff to touchdown before it ever leaves the base. Within reason, there is no time constraint dictating how quickly a path must be generated for the UAV mission. The tradeoff between computational time and accuracy then swings far in favor of accuracy. Path planners of this type attempt to use all available information about mission objectives, enemy locations, and the environment in the computations. They also include better models of the vehicle’s dynamics to ensure path feasibility [22, 23]. A drawback to this group of methods is that they are too slow to deal with flight situations where a UAV would need to re-plan a portion of its path in real time.

The counterbalance to offline path planners is online, or real time, algorithms. Real time path planning is important in allowing a UAV to react dynamically to its environment. Time is very important for these algorithms because a solution must be reached before the UAV encounters the obstacle that necessitated the re-planning. Completeness, especially in terms of dynamics or the number of objectives included in the problem formulation is often sacrificed to meet the time constraint. Some online path planners are designed specifically to re-plan and update a portion of the UAV's flight path that has become undesirable for some reason [24]. Other path planners work by continuously updating or extending the UAV's path in short increments as it flies, which is necessary for missions involving tracking or following other entities [25]. Both groups make important contributions to the field of path planning and a more comprehensive solution to the problem will likely involve coupling these groups in some fashion.

Another method of grouping path planners is by the objectives and constraints included in the cost function formulation. Objectives may be based strictly on path characteristics, such as path length, avoiding enemy missile sites, and passing through reconnaissance areas. Others may incorporate characteristics of the UAV flying the mission. Each UAV has a minimum radius of curvature for turning, an achievable range of flight velocities, and maximum climbing and diving angles that are properties of its design. For path planners that do not incorporate flight dynamics, it is usually assumed that the vehicle will take the path waypoints and fly as close to them as possible given its capabilities. Often, the line between objectives and constraints in path planning optimization problems is blurred. Because unconstrained optimization problems are much simpler to code and solve, many constraints are actually treated as components of the cost function, just like objectives. Formulating the problem in this way is made possible by making the cost of violating a constraint prohibitively high so that a path that violates a constraint will never be an optimal path.

There are a wide range of objectives and constraints included in path planning optimization, as seen in the following two examples. The choice of what to include is based solely upon the opinion of the developer as there is no standard yet for path planning objectives and constraints. One example is a path planner designed to minimize aircraft radar cross section for an unmanned combat aerial vehicle to avoid enemy missiles [26]. Another considers four separate objectives; shortest path, terrain avoidance, minimum and maximum elevation above terrain, and minimum radius of curvature for the path [27]. Even path planners that have similar objectives may formulate their cost functions very differently. Cost function formulation is a very important part of path optimization and there is still plenty of work to be done in determining the best mathematical representation of the problem.

Terrain Information

One important constraint peculiar to UAV path planning is terrain avoidance. It is generally approached as a specialized version of obstacle avoidance. Avoiding terrain is more complicated than avoiding obstacles because there is a vastly larger amount of information involved. Using terrain data in path planning has been made possible by the rapidly growing availability of digital elevation data and computer codes that are able to make use of it.

Many applications now take advantage of terrain information, with earth sciences leading the way. For example, the National Geophysical Data Center is currently building detailed elevation models for specific US coastal regions to better model and predict tsunamis [28]. Others are investigating how terrain data can be used to determine soil moisture levels and the data point resolution required to get good results [29]. The field of landscape architecture is making use of digital terrain data and the increasing power of computer game engines to render and manipulate real worksites virtually [30]. Other design researchers are creating techniques to piece together bits of real terrain data to create realistic virtual landscapes [31]. In the area of robotics and path planning, terrain data has become an important component of computing feasible or alternative paths for unmanned vehicles [32].

Terrain data is presented in a variety of formats. Originally elevation data was stored in ASCII text files. While it is human-readable, it uses a great deal of memory and takes the computer longer to process. Now a wide variety of formats exist for storing terrain elevation data in a more efficient manner. These formats range from binary data encoding to proprietary data structures, depending on who captured the elevation information. The National Oceanic and Atmospheric Administration of the United States produces the most widely available formats. One format, the USGS Digital Elevation Model (DEM) data set, simply stores a regular rectangular grid of height values. The point separation can range between 10 meters to 90 meters or more depending on the data set. DEM data has only been generated for the United States and can be downloaded from the Internet. A second available format is the Digital Terrain Elevation Data (DTED). DTED was originally collected for military use, but since then some of it has been made public. There are several levels of accuracy with level zero having data points separated by 1000 meters and level two having points separated by 30 meters [33]. For worldwide data, there are several possibilities including the Global Land One-km Base Elevation Project [34]. Generally, high-resolution terrain data is much harder to find for areas outside of the United States.

This has given rise to many different programs that read in the various terrain data formats and then display it for the user or allow the user to manipulate it. Some software is available free online such as GRASS or the ArcGIS viewer. Other software, like MetaVR, or the Unreal Engine are quite expensive but offer much higher graphics capabilities. Then there are libraries like GDAL that are designed to take a wide variety of terrain files and allow other applications to load them even if there are differences in formatting.

The use of terrain data in path planners has grown as the quality of the available data has improved. Figure 8 provides an example of a path planner that uses unrealistic, mathematically generated terrain (colored by height) [35] and one that uses actual terrain data in planning [36]. Sinopoli et al. have developed a vision based path planner that uses a coarse terrain representation for global planning and a refined grid for local evaluation to allow the UAV to fly close to the terrain while avoiding obstacles [36]. Another path planner developed by Mittal and Deb seeks to optimize the competing goals of minimizing path length and maximizing safety, considering terrain and ground obstacles [35].

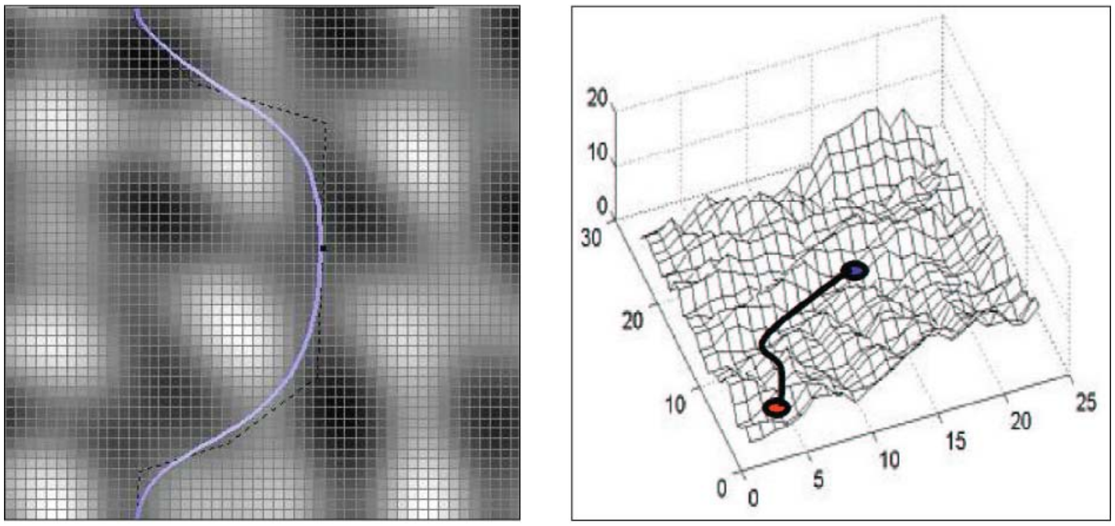


Figure 8. Mathematically generated height field (left) and actual terrain (right) used in path planning

Yet, even as terrain information is more widely used in path planners, many employ mathematically generated height fields and not physical terrain data. It is simply assumed that an algorithm that performs well on simulated terrain will also perform well on actual terrain. A second concern is that using simulated terrain bypasses the non-trivial issues of obtaining the terrain information for the area in which the UAV will be conducting its mission and allowing the path planning algorithm to access the needed data in an efficient manner.

User Interaction with Path Planner Output

A path generated by a path planner is spatial in nature and lends itself naturally to visualization in a 3D environment as mentioned earlier. It would make sense then that an operator would be far more adept at analyzing a path presented visually, with other relevant features such as terrain or enemy-unit locations also displayed, than textually. The operator should be able to gauge quickly how well the algorithm performed. Performance in this sense is determined in the individual's subjective opinion of how well the path meets the given objectives and constraints. A path planner that consistently produces results that fit well with the operator's intuition would inspire confidence on the part of the operator. Algorithms that perform poorly will cause distrust on the part of the operator. As the operator has less direct control over the actions of a UAV, it is important to establish a high level of trust in what the algorithm is doing behind the scenes.

Even if the operator trusts the path planner, it would be valuable to allow the individual to tailor the output of the algorithm to better match personal preferences and mission specific considerations. Regardless of how simple or complex an algorithm is in terms of the number and variety of constraints it handles, the operator often has no means of incorporating preferences into how those objectives are handled. For low-priority missions, the most important objective may be the safety of the UAV, while, in a covert operation, flying as close to the terrain as possible to avoid radar detection may be more important. Without an operator to alter the relative importance of the objectives in some sense, the output of the path planner will be sub-optimal.

Almost all path planners return one alternate path to the operator. This assumes that the best alternative found by the optimization algorithm is at least an acceptable path, if not the optimal path, and will be used. There is no alternative course of action for the operator if the alternative path does not satisfy the operator. If a path planner took into account all the possible objectives and constraints for the UAV and the mission and all the preferences of the operator, automatically implementing the top alternate path would not be a problem because the algorithm would choose correctly every time. Such a path planner does not yet exist. It would be very valuable then to allow the operator to have more control and more options when decisions need to be made on re-tasking UAVs.

Decision Making

Decision making forms an important aspect of UAV path planning as a human presence is included in the path selection process to validate proposed actions of the UAV. The path planner operation and output must be designed to facilitate quick and accurate interpretations of the data so that the UAV controller can choose the appropriate alternative under the tight time constraints of a combat situation. Knowing how individuals in combat situations handle information should influence what information is presented and how it is done. When military officers encounter such situations, decisions are generally made by creating a vision for the mission outcome, generating a plan which is evaluated and refined, and finally issuing the specific orders to carry out that plan. Most frequently, it is intuition based on experience that leads to the best action plan [37]. The path planner generates one or multiple alternate paths, relieving the operator of this task, and allowing that person to focus on evaluating the paths in light of the mission objective and choosing the best alternative. More alternatives will increase the chances that the operator will intuitively recognize a path which fits very closely with the mission objective.

Intuition and past experience form a core for how the decisions are made. The three stages involved in this process are: model construction, revision, and falsification. Stage one involves making a model of the situation. In the following two stages the user seeks out more detailed information to come to a preliminary conclusion which is accepted if it is not falsified by further information [38]. Such a thinking process would lend itself well to a tiered information display approach where an attractive group of solutions is selected first, followed by choosing one of the individual solutions from that group upon completing a more detailed investigation.

Information display techniques can significantly influence the operator's ability to make decisions correctly. The operator will be biased largely toward interactive visual information (which is absorbed most quickly) and less toward textual information (which is processed more slowly) in decision making processes [39]. Proper path display will leverage this bias by giving key information, such as path optimization criteria and waypoint positions, the most visual weight.

More information facilitates better decision-making up to the point where the operator experiences information overload and can no longer process it quickly. Several tactics including overview + detail, focus + context and information hiding are commonly employed to prevent information overload as a user investigates the data [40]. Utilizing these types of techniques would allow a user to evaluate more alternatives or the same number of alternatives in more detail in the same amount of time and make a better-informed decision.

Most research on decision making in path planning takes the approach that the goal is to eliminate completely the need for human input. Methods of this form include, for example, a dynamic path planning algorithm that continuously plans short path sections [41] and a hierarchical planning structure to develop an overall solution which is refined as path sections are analyzed in increased detail [42]. In the future, it could be possible to eliminate completely human input from UAV command and control, but the field will need to mature significantly before combat vehicles will be allowed to operate with this degree of autonomy.

The other approach is based on the concept that there can be multiple good alternate paths for a given situation. One algorithm incorporating multiple paths was designed for dynamic environments using cluster analysis with an evolutionary algorithm [43]. The motivation behind its design was to prevent the planner from needing to recalculate if the dynamics of the situation changed because there would still be other alternate paths available. The operator would have the ability to analyze the dynamics of the situation and then select a path accordingly. While the concepts are good, there is still much work to be done in improving the implementation of a multi-path approach, especially in path visualization.

Path Visualization

Visualization and information display are a critical part of the path planning process as they link the operator to the UAV. Most broadly, these links include video feeds, sensor feeds, system status updates, vehicle statistics (like flight speed and altitude), and environmental conditions which are presented to the operator in some fashion. Path planning is no exception. It is necessary for the operator to see and understand where the UAV is and where it will be going.

Visualization is a weak area in path planning as the focus in this field leans strongly toward algorithm development. It seems that many researchers are looking to the goal of a perfect path planner where a human-in-the-loop is no longer necessary. In that situation, visualization becomes a courtesy rather than a necessity. As a result, path representation and effective visual communication of information to the operator is given very little thought. This is an area of path planning in need of focused exploration and development.

There are a wide range of visualizations used with path planners. The ones described below present a broad sampling of what is currently being done. Some path planners offer strictly 2D interfaces [44]. Figure 9 presents the output of an obstacle avoidance path planner for an urban type environment. Notice that the operator has no indication on the plot of the height of the path, which is a constant established somewhere within the algorithm, or of the height of the obstacles. The representation in Figure 10 is for a small UAV path planner that takes into account heading and wind direction [45]. Again, the height is simply set at a constant for the entire path and not listed in the display.

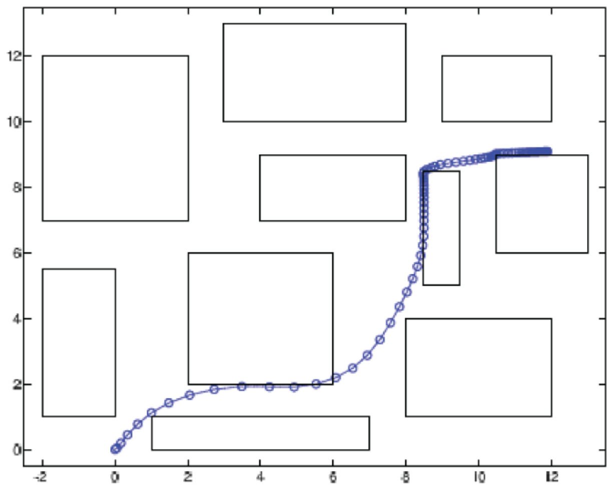


Figure 9. 2D representation of a UAV path in an urban environment

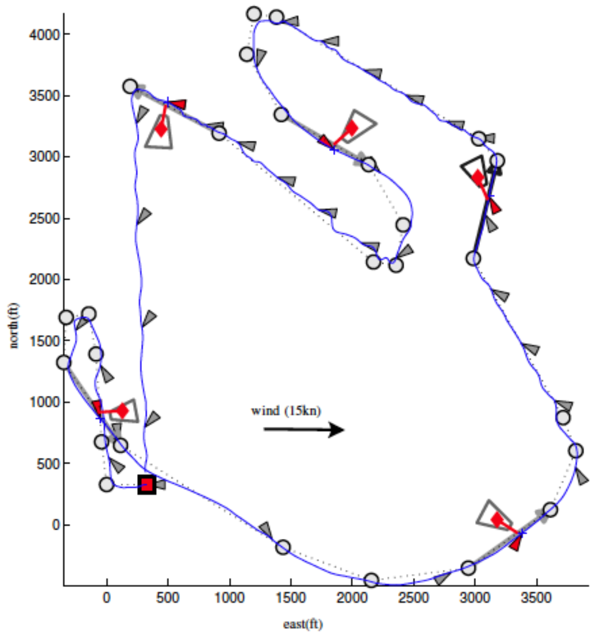


Figure 10. Path at constant altitude accounting for heading and wind direction

A slightly more complex category of visualizations include path planners with split 2D windows for presenting the path to the operator [46]. In Figure 11, there are two graphs.

The top graph shows the path trajectory from the perspective of a person looking straight down on it. Altitude and time are plotted on the bottom graph. The two graphs present UAV 3D location and time, but it is very difficult to couple the information because the operator is not given any indication of the relation between position and time.

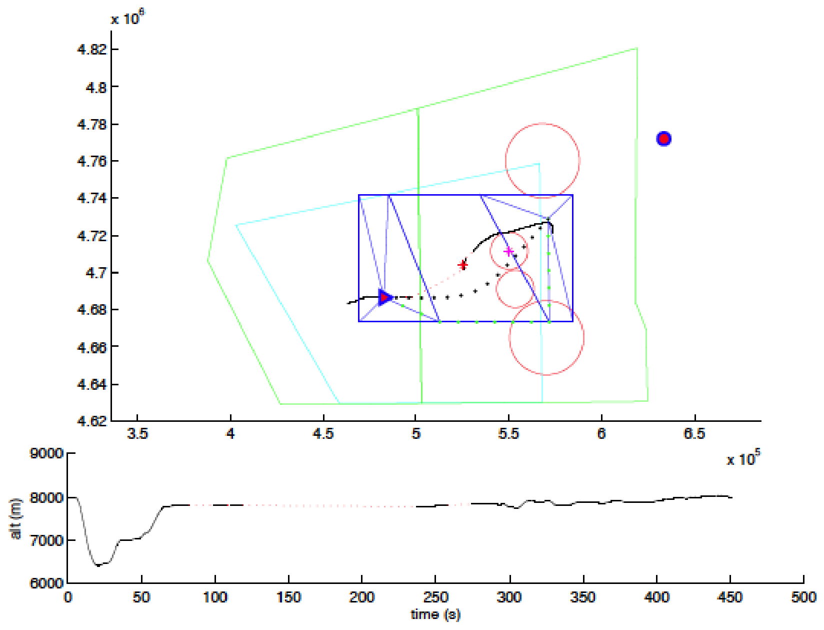


Figure 11. Path representation with two 2D views

The next level of path visualizations makes use of a single isometric view to convey information to the operator. Figure 12 and Figure 13 give two examples of this type of display [47, 48]. Isometric views allow the operator to build a better 3D mental model of the paths. A 3D view is helpful, but is limited in that the operator cannot interact with it or change the established viewing location. To date, very little has been done as to developing high quality visualizations that allow the operator the freedom to explore path alternatives in any way desired.

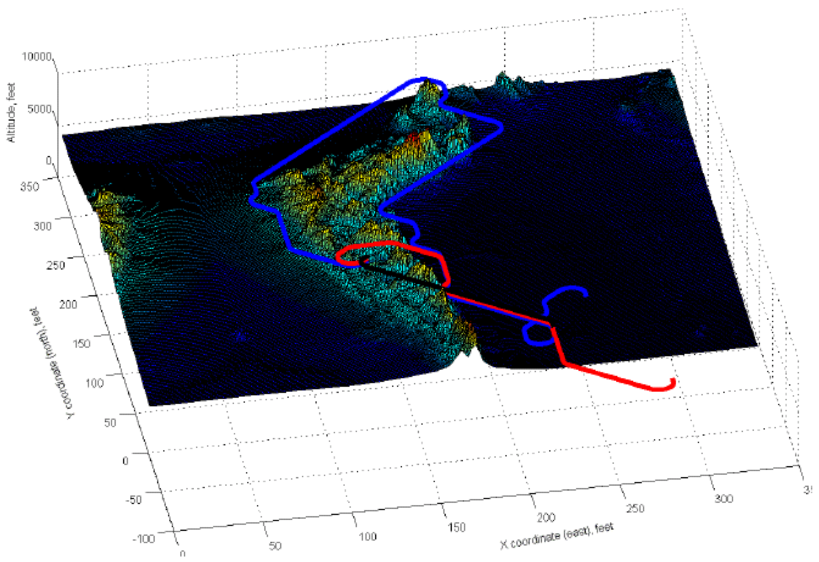


Figure 12. Isometric view of several paths with terrain shown

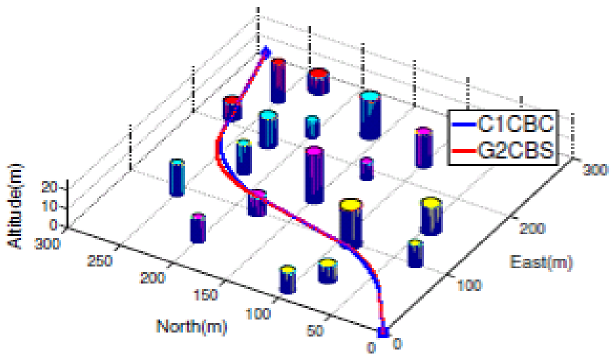


Figure 13. Simpler visual representation of a path from an isometric view

Visualizations can also be categorized by how many paths they present to the operator. In all the figures above, the operator had one alternative (though some graphics do show more than one path) and was not involved in the planning process. There are very few path planners that offer the operator alternative paths to chose from. Unfortunately, while the fact that the operator is making a decision should make visualization of the alternative paths more important, the representations are still very simple. Figure 14 shows a path planner for dynamic environments where the operator can choose between two alternate paths based on personal preference [43]. The path planner in Figure 15 takes a slightly different approach [49]. The original path remains an option with a safe and moderately safe alternative for each pop-up threat. Depending on how risky the operator chooses to be, one of the three alternatives can be selected.

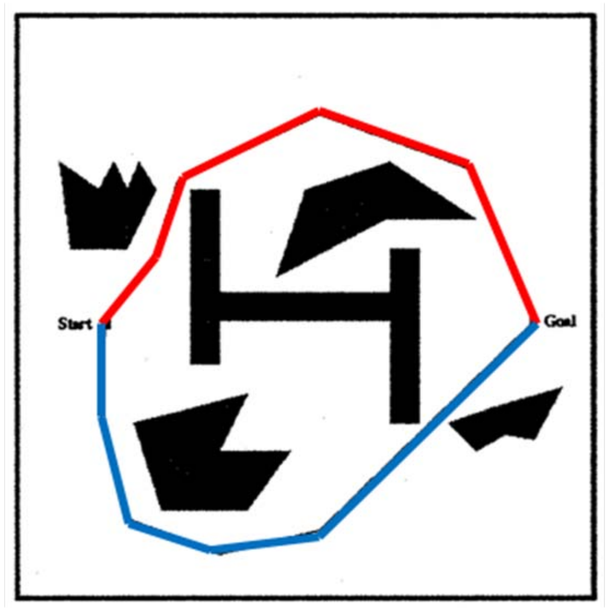


Figure 14. Path planner with 2 alternate paths

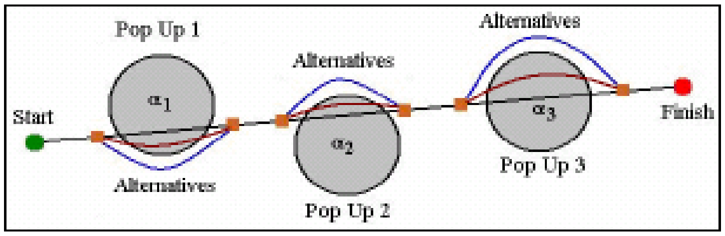


Figure 15. Path with multiple alternatives for each threat

Summary of Current Path Planner Research

Research is being done on all facets of the path planning problem. Many path planners have been developed that contain part of the solution to the problem, but the challenge remains to create a complete package for use in UAV operation and operator training. The major components of a path planner should be at least: a complete set of objectives (accurately modeled) for the UAV mission, an optimization algorithm to generate optimal paths according to those objectives, a clear visual representation of the algorithm output, and a method for the operator to interact with that output. Accordingly, the path planner presented in this thesis will focus largely on these areas in an effort to continue moving toward a more complete solution to the path planning problem.

Research Issues

Based on the literature survey and the conclusions drawn from it, two research issues have been identified:

1. *Improving the cost function by incorporating terrain information and improved mathematical models for reconnaissance gathering and threat avoidance.*

   The objectives included in the optimization cost function and their formulations were reworked to provide a higher fidelity representation of the UAV's capabilities and those of enemy threats. Terrain avoidance was added to prevent infeasible paths from being presented to the operator. The goal is to present high-quality solutions to the operator in real-time to allow more efficient re-tasking of the UAV.

2. *Develop a visualization strategy for multiple paths to enable improved operator decision-making.*

Returning multiple paths allows the operator to decide on the best alternate path for re-tasking the UAV. How the paths are visualized and the information that is provided about the paths will impact the operator’s ability to absorb the necessary information quickly and make a decision based on mission objectives and personal preferences.

Chapter 3 – Method Development

Original Algorithm

The general approach for performing path planning for UAVs is outlined in Figure 16. The base algorithm was enhanced and modified to increase the fidelity of the simulation, to investigate different visualization techniques, and to develop different interaction methods. These enhancements (not the original algorithm) comprise the work presented in this thesis. This research uses the Virtual Battlespace as a platform for investigating path planning.

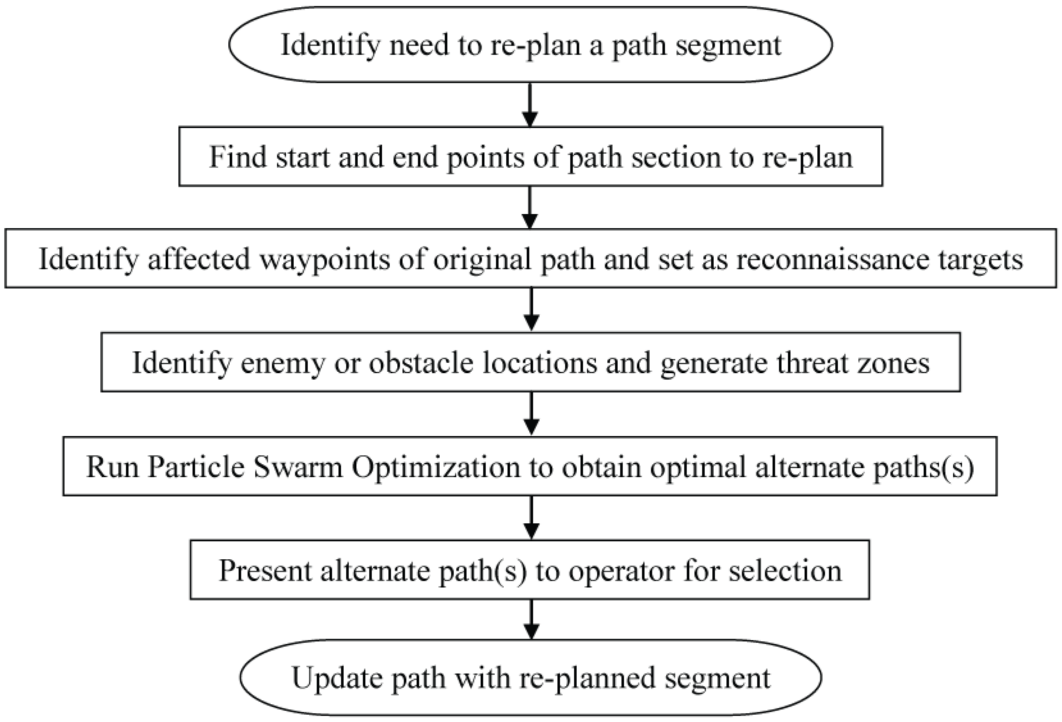

Identify need to re-plan a path segment
Find start and end points of path section to re-plan
Identify affected waypoints of original path and set as reconnaissance targets
Identify enemy or obstacle locations and generate threat zones
Run Particle Swarm Optimization to obtain optimal alternate paths(s)
Present alternate path(s) to operator for selection
Update path with re-planned segment

Figure 16. Flowchart of the path planning process

The path planning process begins when a UAV identifies something in the environment that requires a change in its original path. At that point, the Virtual Battlespace will pop up an alert, as shown in Figure 17, notifying the operator of the situation. The operator can choose to ignore the alert (and not re-task the UAV) or investigate the alert (and enter the path planning mode of the Virtual Battlespace). This ensures that as the operator is overseeing the operation of multiple vehicles, important situations that require human intervention are not overlooked. Figure 18 shows the original path with three alternate paths and the red threat dome (represented by the red circles) they are attempting to avoid. The yellow fence is the original path of the UAV.

Figure 17. Alert message in Virtual Battlespace

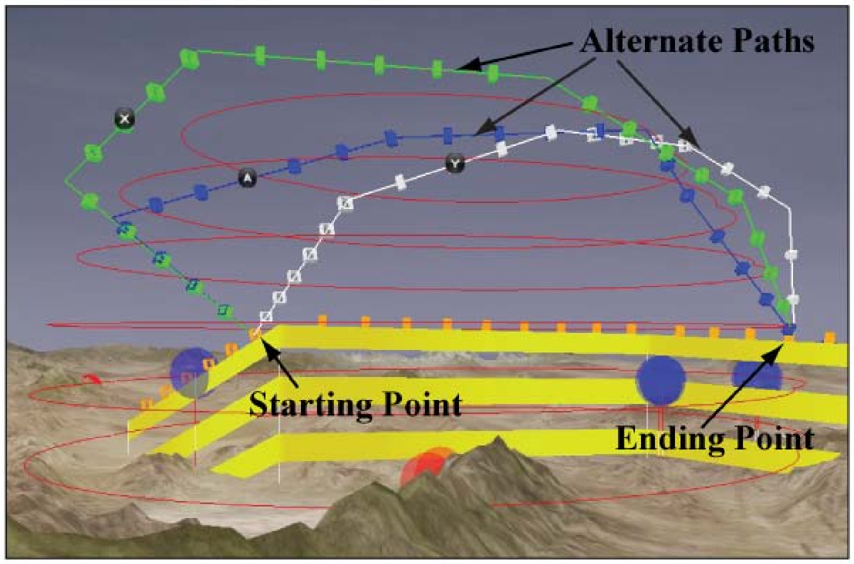


Figure 18. Diagram of alternate paths with labeled features

The first step of the path planning process is to identify the starting and ending points of the path segment to be re-planned. The starting point is set on the original path by projecting the position of the UAV forward in time a user-defined amount, as seen in Figure 19. That lead time (by default set to 30 seconds) determines how much time the operator has to re-task the vehicle. The time starts when the operator decides to plan an alternate path and therefore includes the time the Particle Swarm Optimization algorithm takes to generate its solutions.

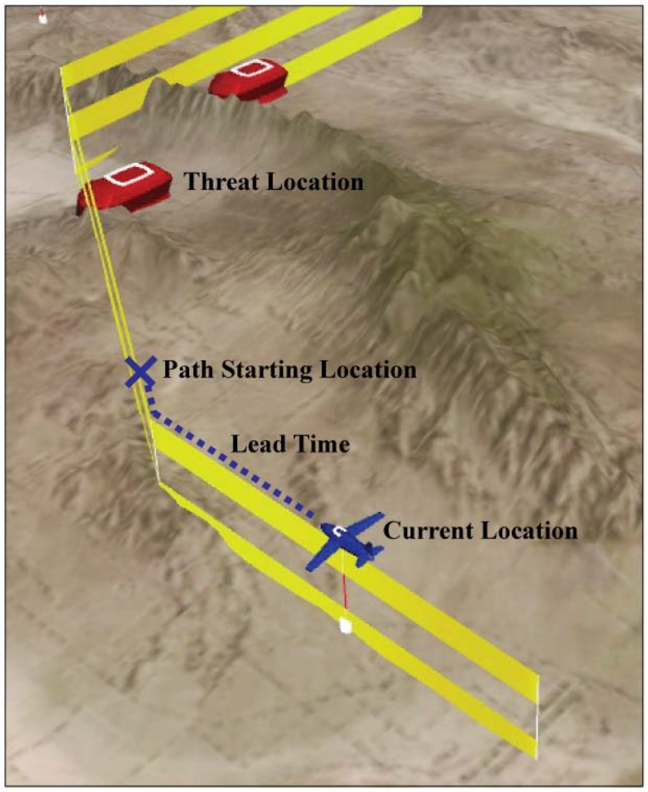


Figure 19. Diagram for determining path starting location

The ending point is found by a simple test: A search algorithm steps along the original path and, at each step, checks to see if that location is safe (outside of any threat zones). The first safe location is set as the ending point for the segment to be re-planned. As the algorithm moves along, it also searches for entities within a user-specified range of the path and stores their identities for future evaluation.

For the second step, any waypoints located between the starting and ending points are stored as reconnaissance targets. The third step is to identify threat locations and generate threat zones. A threat zone is currently represented by the red lines as a hemisphere such that the UAV is considered safe from the threat if it is outsize that zone. The entities stored earlier by the search algorithm are evaluated and identified as either enemy locations (that threaten the UAV) or friendly locations (that should be avoided to prevent collisions). Their locations and identities are stored as well.

All of this information (the starting point, ending point, reconnaissance targets, and threat locations) is then passed to the PSO path planner. The PSO algorithm takes in the data, generates optimal alternate paths based on user-defined objectives, and returns those paths to the Virtual Battlespace environment for visualization. Path generation and path visualization are decoupled problems, allowing each to be investigated independently. The work done on the optimization algorithm will be presented first, followed by the work done on visualization and decision making.

Figure 20 is a simple 2D illustration of a 3D threat avoidance problem. The green line is a segment of the UAV's original path. The black dot is an unexpected threat (located at Z_T) with the red circle representing the threat zone (also referred to as a threat dome because of its shape). For enemy vehicles, the red area is modeled as a sphere (a hemisphere for ground vehicles) with radius R_T. To avoid collisions with friendly vehicles, they are also modeled as threats, but with a smaller threat zone radius, R_F. R_T and R_F are user-defined values set to 20,000 feet and 2,000 feet respectively. The blue circles (centered at locations Z_R) are reconnaissance zones (modeled as vertical cylinders) with radius R_R set to 1,000ft.

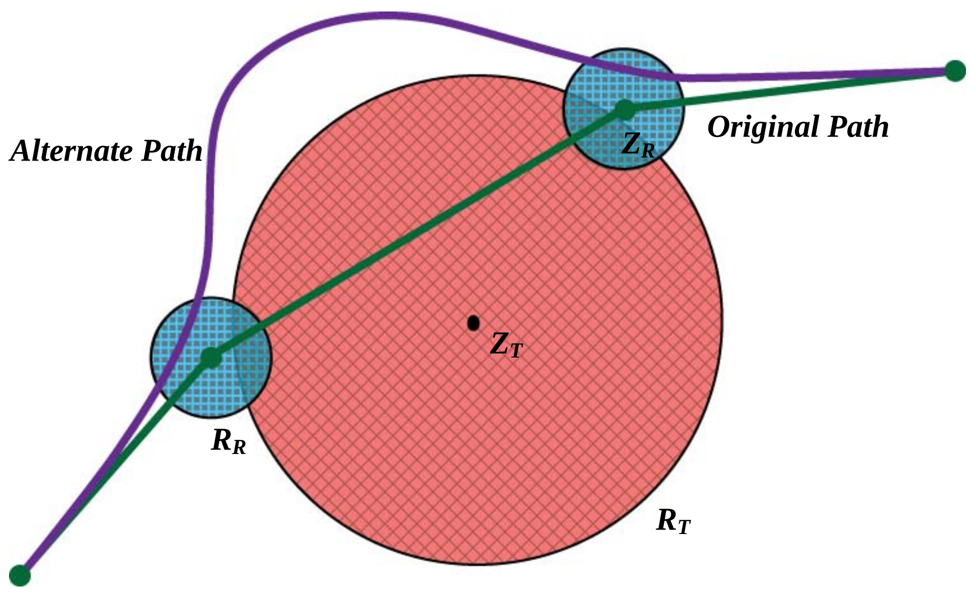


Figure 20. Two-dimensional illustration of a simple threat zone avoidance problem

Since the green line passes within range of the unexpected threat, the threat would trigger the path planning process. The green points (the starting point, ending point, and affected waypoints) are the initial design points for the problem. From these points, a 3D design space is created which will be searched to find an optimal alternate path. The size of the search space needs to be properly sized for efficient operation of the PSO algorithm. Having a space that is too large will be computationally expensive to investigate. Having a space that is too small may prevent good alternate path choices from being considered.

In this simple path planning illustration, the goal is to have the UAV pass through the reconnaissance zones while avoiding the threat zone. The purple alternate path is a representative solution the algorithm might generate to meet this goal. PSO works to find the best compromise between objectives (based on their importance) as it searches for the optimal path.

Original Particle Swarm Optimization Formulation

PSO begins by randomly initializing a set of swarm members within the design space. Each swarm member is an alternate path modeled as a B-spline curve. The size of the swarm is defined by the user. Having a larger swarm allows for more exploration of the design space per iteration but also increases the computational cost.

Optimality is determined by the cost function; paths with lower computed fitness values are more optimal than those with higher fitness values. Each path is broken into short line segments and these segments are evaluated against the cost function. This can be seen in Figure 21 where *p(u)* is the approximation of the B-spline curve for the alternate path and *u* is the set of segments making up that approximation (each with an x, y, and z component). The original path $p_0(u)$ can also be broken into line segments for computational analysis, though this is done only once because that path is static. As the number of path segments increases, the accuracy of the approximation improves but the number of computations per path also increases.

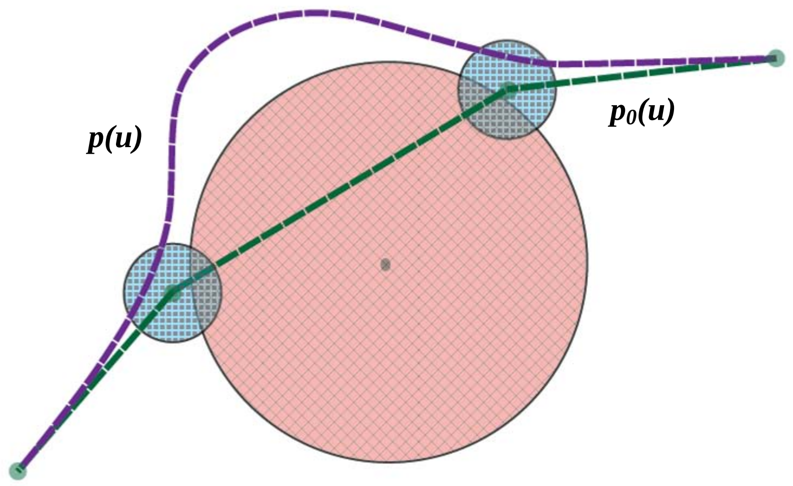


Figure 21. Alternate path B-spline curve broken into line segments for evaluation

Once the path is broken into line segments, the fitness of that swarm member can be evaluated according to the cost function. Formulation of the optimization cost function involves selecting the objectives to include and then expressing those objectives in a mathematical form that can be used to evaluate the alternate paths. Three objectives are incorporated into this path planner: fuel efficiency (C_F), reconnaissance (C_R), and threat avoidance (C_T). The cost function, with one component for each objective, is as follows:

$$C = K_1 C_F + K_2 C_R + K_3 C_T \tag{4}$$

The constants K_1, K_2, and K_3 in Equation 10 are component weights that determine the relative emphasis placed on each of the objectives in the cost function. Weights are assigned by the operator *a priori* to align the relative importance of the objectives with personal preferences and the mission briefing. Table 1 shows an example of three different weights that will emphasize fuel efficiency, reconnaissance, and threat avoidance, respectively, in the solutions that are generated. The original cost function formulation [50] and the modifications will be discussed for each component in turn.

Table 1. Example of component weights used to generate a set of alternate paths

	Fuel Weight, K_1	Recon Weight, K_2	Threat Weight, K_3
Fuel efficiency path	0.60	0.20	0.20
Reconnaissance Path	0.20	0.60	0.20
Threat Avoidance Path	0.20	0.20	0.60

B-splines

The alternate paths are modeled as B-splines to simplify the computations. Figure 22 shows a 2D B-spline with its control points. As little as three control points can be used to define a simple parametric curve. B-splines have been successfully used in modeling constrained curves, accommodating the necessity of having the starting point and ending point of the alternate paths fixed at locations on the original path, as described earlier [51]. A brief, general introduction to B-splines is included here to provide a better understanding of how they are used in the path planning optimization problem formulation. Geometric modeling textbooks, such as the one by Gordon and Riesenfeld contain a more complete description of B-splines [52].

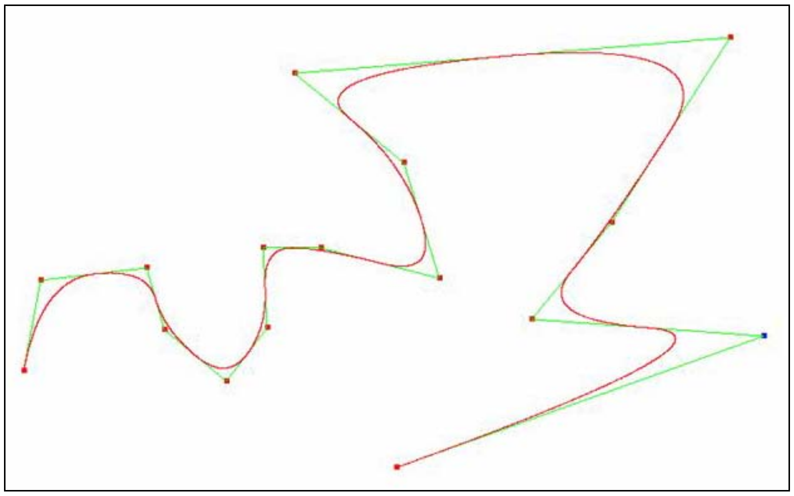


Figure 22. B-spline (red curve) with control points (red dots)

A parametric B-spline *p(u)* of order *k* (degree *k-1*) is defined by *n+1* control points $p_i(u)$, knot vector *x*, and the relationship:

$$p(u) = \sum_{i=0}^{n} p_i N_{i,k}(u) \quad \textbf{(5)}$$

The Bernstein basis functions, *N*$N_{i,k}(u)$, are generated recursively using

$$N_{i,1}(u) = \begin{cases} 1 & \text{if } (x_i \le u \le x_{i+1}) \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$N_{i,k}(u) = \frac{(u - x) N_{i,k-1}(u)}{x_{i+k-1} - x_i} + \frac{(x_{i+k} - u) N_{i+1,k-1}(u)}{x_{i+k} - x_{i+1}} \tag{7}$$

The n-1 control points, along with the degree of the B-spline determine its shape. To understand the shape of the B-spline, a polyline can be created that connects all the control points in order. A third degree B-spline (as is used in this path planner) will stay as close to that polyline (and therefore the control points) as possible.

Improving the Original Path Planner

Based on the research issues presented in chapter two, the following improvements have been implemented:

1) Both the reconnaissance cost and the threat avoidance cost were modified to better model the data gathering abilities of the vehicle and the likelihood of the UAV to be hit by an enemy.

2) Terrain collision avoidance was added as an additional cost to prevent infeasible paths from being returned to the operator as optimal solutions to the problem.

3) At the same time, the cost values for fuel efficiency, reconnaissance, and threat avoidance were all scaled to be of the same order of magnitude (to be between 10 and 100). This

scaling should aid PSO in generating optimal solutions that correspond well with the weights that were set for the objectives.

4) Meta-paths were then created for visualizing the alternate paths to aid the operator in the decision making process.

Re-initialize Swarm Members

Before starting a new iteration, one additional procedure is conducted: all the paths are checked against *gBest* and those that are functionally identical are re-initialized. This is a two-tiered test designed to force increased exploration of the design space with the goal of finding as many locally optimal solutions as possible as well as the globally best solution.

The first step of the path similarity test is a comparison of the fitness value for swarm member *n* ($fitness_n$) to the *gBest* fitness ($fitness_{gBest}$) as in Equation 7. If the difference is less than 0.99 (the user-defined value of C_V), the paths are considered distinct and the swarm member is unchanged. If it is not, the paths are compared on the basis of similarity in 3D space according to Equations 8 and 9. The square of the distance between segment *i* on *gBest*, at location $[p_{gBest}(x_i), p_{gBest}(y_i), p_{gBest}(z_i)]$ in 3D space and the corresponding segment *i* on path *n*, denoted as p_n (also in 3D space) is computed and then summed over all *N* path segments. If the separation value D_n is less than a user-defined constant C_I, the paths are considered identical and the swarm member is reinitialized. If not, nothing is done.

$$V = \begin{cases} true, & \left| \dfrac{fitness_n}{fitness_{gBest}} \right| < C_V \\ false, & \left| \dfrac{fitness_n}{fitness_{gBest}} \right| \geq C_V \end{cases} \tag{8}$$

$$D_n = \sum_{i=1}^{N} \left(p_{gBest}(x_i) - p_n(x_i)\right)^2 + \left(p_{gBest}(y_i) - p_n(y_i)\right)^2 + \left(p_{gBest}(z_i) - p_n(z_i)\right)^2 \tag{9}$$

$$I = \begin{cases} true, & D_n < C_I \\ false, & D_n \geq C_I \end{cases} \tag{10}$$

The first check is used to eliminate unnecessary calculations by ruling out dissimilar paths immediately. The second check is needed because simply having the same fitness value does not guarantee similarity in a multi-modal problem. The comparison is performed for all swarm members. Following the path similarity check, the next iteration begins by breaking each of the new B-spline paths into line segments for evaluation against the cost function.

Fuel Efficiency Component Cost

Original Formulation

Because vehicle dynamics are not incorporated into this path planner, the fuel efficiency cost component C_F is based strictly on path length. As fuel consumption depends on the vehicle, speed, rate of climb or descent, and other environmental factors, approximating it with path length is a gross simplification, but it forms a starting point for future work. The value of C_F

is modeled as the additional distance the UAV must travel if taking the alternate path instead of the original path:

$$C_F = L - L_0 \ , \qquad L = \sum_{i=1}^{N-1} \left| p\left(u_{i+1}\right) - p\left(u_i\right) \right| \tag{11}$$

The curve length of the path, *L*, is the sum of the lengths of all the line segments $p(u_i)$ that approximate it. L_0 is the length of the original path, computed in the same manner. Since L_0 may not be the shortest distance between the starting and ending points, it is possible to have an alternate path that is shorter than the original path and a corresponding negative value for C_F. This cost increases linearly with path length.

Modifications to the Original Formulation

Essentially the only modification to the fuel efficiency component has been scaling the range of values it can assume. Initially, the cost was revised to fall between zero and one. To make this possible, L_0 was redefined to be the distance in 3D space from the starting point to the ending point of the segment being re-planned. The cost function was then:

$$C_F = 1 - \frac{L_0}{L} \tag{12}$$

The behavior of this function tends to heavily favor shorter paths and makes it hard to balance path length against other objectives to find locally optimal solutions. To correct this behavior, the fuel efficiency cost was modified to be linear, as shown in Equation 13. The smallest possible value is 10 and the cost increases uniformly with increasing length.

$$C_F = 10\left(\frac{L}{L_0}\right), \qquad L = \sum_{i=1}^{N-1}\left|p\left(u_{i+1}\right) - p\left(u_i\right)\right| \tag{13}$$

Reconnaissance Component Cost

Original Formulation

The second objective is reconnaissance. It is assumed that the original path waypoints (set as reconnaissance targets) are the optimal locations for viewing that target. It is also assumed that there is an area around the waypoint that will allow the UAV to observe the target, but the quality of the gathered data degrades as the UAV gets farther from the target. Equation 14 is the reconnaissance component cost calculation for one path segment *p(u$_i$)*.

$$d\left(p\left(u_i\right), Z_R\right) = \min\left[\left(\text{Distance between point } p\left(u_i\right) \text{ and } Z_{R,j}\right) - \left(R_R\right), \quad j = 1, 2, \ldots, N_R\right] \tag{14}$$

The distance from that segment to each of the *j* reconnaissance targets $Z_{R,j}$ is found (minus the radius of the reconnaissance zone (R_R). The distance is calculated in the horizontal plane only; height is not included in this formulation (which means a path that passes 1,000 feet above the target is the same as one that passes 10,000 feet above it). If the value is greater than zero, it is added to the total reconnaissance cost, C_R, in Equation 15. This calculation is repeated for all *N* path segments. No cost is incurred if the UAV passes inside the reconnaissance zone but the cost increases linearly as the vehicle strays from that optimal viewing zone.

$$C_R = \sum_{i=1}^{N} C_i, \qquad C_i = \begin{cases} \left|d\left(p\left(u_i\right), Z_R\right)\right|, & d\left(p\left(u_i\right), Z_R\right) \geq 0 \\ 0, & d\left(p\left(u_i\right), Z_R\right) < 0 \end{cases} \tag{15}$$

Modifications to the Original Formulation

The modifications to the reconnaissance cost were made to improve the fidelity of the model. Instead of looking at just distance from the target, now the cost is a function of viewing angle. As seen in Figure 23, the target zone occupies some field of view, γ, from the perspective of the UAV and its sensors. It is assumed that as the angle γ gets larger, the UAV will be able to gather data on the target, Z_R more effectively.

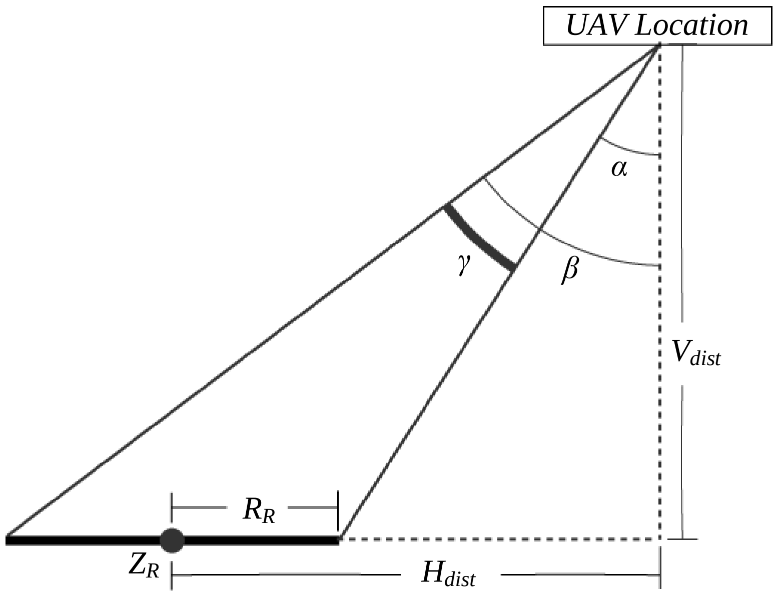


Figure 23. Diagram of variables needed for computing the reconnaissance cost

Based on the distances H_{dist} and V_{dist} and the reconnaissance zone radius R_R, the field of view could be calculated. This model was overly complicated as distance and angle are adequate to provide an indication of how well the UAV can view a target. Figure 24 encapsulates the concept driving the new cost formulation. The optimal viewing position is assumed to be directly above the target (small angle) and as low as possible (small distance). If distance is too large, the target is very small from the perspective of the sensors, making it difficult to gather data on the target. Also, if the angle is too large, the target is very thin, leading to the same problem.

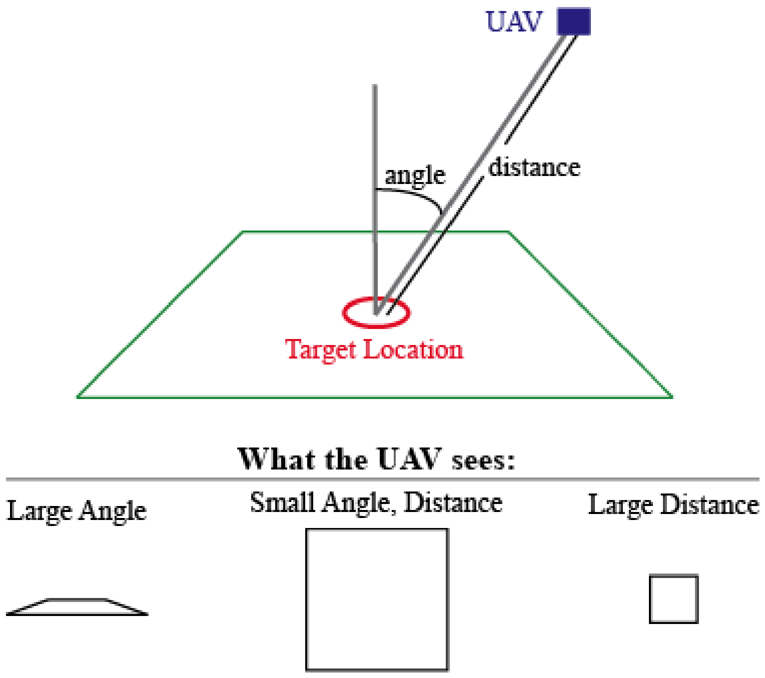


Figure 24. Key parameters for reconnaissance cost and their effects

A cost is calculated for each reconnaissance target Z_R for each path segment $p(u_i)$ with two components: distance and direction. The distance value is the separation between the UAV path segment $p(u_i)$ and the target Z_R in 3D space, scaled by the reconnaissance zone radius, R_R as shown in Equation 16. The angle, θ, is simply the inverse tangent of the horizontal distance over the vertical distance as seen in Equation 17. For each reconnaissance target, the minimum value produced by any path segment $p(u_i)$ was kept, according to Equation 18. These values were summed and then scaled by 10 over the number of reconnaissance targets N_R to produce the reconnaissance cost component (see Equation 19). Here is a case where it is possible for the cost to be less than 10 if the alternate path passes very close to all the original waypoints, though in practice that small of a cost is rare. A path with a reconnaissance cost this low would most likely make it the optimal path for that optimization run but would not prevent the algorithm from finding other locally-optimal paths as well.

$$d\left(p\left(u_i\right), Z_R\right) = \frac{\text{Distance between points } p\left(u_i\right) \text{ and } Z_R}{R_R} \tag{16}$$

$$\theta_i = \tan^{-1}\left(\frac{H_{dist,i}}{V_{dist,i}}\right) \tag{17}$$

$$C_j = \min\left(\theta_i + d\left(p\left(u_i\right), Z_R\right),\ \ i = 1, 2, \ldots, N\right)\ , \qquad j = 1, 2, \ldots, N_T \tag{18}$$

$$C_R = \frac{10}{N_R}\sum_{j=1}^{N_R} C_j \tag{19}$$

Threat Avoidance Component Cost

Original Formulation

Avoiding threat zones (and thus hopefully preventing the loss of the UAV) is the third objective of the path planner. Calculating the likelihood of an enemy being able to damage a UAV involves modeling the threat's effectiveness as a function of the UAV's position relative to the threat. A threat is modeled as a dome with the cost increasing linearly from zero at the outer edge to R_T at the location of the threat. For each path segment $p(u_i)$ and each threat Z_T, the separation distance in 3D space is found and subtracted from the threat zone radius. The threat zone violation $d(p(u_i),Z_T)$ represents how far inside the threat dome the path segment is located. The threat avoidance cost is then the sum of the threat zone violations over all the path segments (where a given segment can incur multiple costs if it violates multiple threat zones. No cost is incurred if the path stays outside of all threat zones. The formulation of the threat avoidance cost component is given in the following equations:

$$d\left(p(u_i), Z_T\right) = (R_T) - \left(\text{Distance between point } p(u_i) \text{ and threat } Z_T\right) \tag{20}$$

$$C_T = \sum_{i=0}^{N-1} C_i \ , \qquad C_i = \begin{cases} \left|d\left(p(u_i), Z_T\right)\right|, & d\left(p(u_i), Z_T\right) \geq 0 \\ 0, & d\left(p(u_i), Z_T\right) < 0 \end{cases} \tag{21}$$

Modifications to the Original Formulation

The threat avoidance cost was first scaled to be in the same range as the other cost components (so that the cost is between 10 and 100). This was done as follows:

$$C_T = 10 + \left[\frac{1}{N_T}\right] \sum_{i=1}^{N} \sum_{j=1}^{N_T} C_{i,j} \ , \qquad C_{i,j} = \begin{cases} \dfrac{d\left(p(u_i), Z_{T,j}\right)}{R_T}, & d\left(p(u_i), Z_{T,j}\right) \geq 0 \\ 0, & d\left(p(u_i), Z_{T,j}\right) < 0 \end{cases} \tag{22}$$

Threat avoidance was then modified a second time to investigate a non-linear approximation of the risk to the UAV within a threat zone. The distance between the UAV and the threat, $D_{T,j}$ for threat *j*, is still the input parameter. Now it is handled as in the following equation:

$$C_T = 10 + \left[\frac{1}{N_T}\right] \sum_{i=1}^{N} \sum_{j=1}^{N_T} C_{i,j} \ , \qquad C_{i,j} = \begin{cases} 1 - \dfrac{\sqrt{D_{T,j}}}{\sqrt{R_T}}, & \mathrm{D}_{T,j} < R_T \\ 0, & \mathrm{D}_{T,j} \geq R_T \end{cases} \tag{23}$$

The square root is what makes the cost non-linear. The threat avoidance cost $C_{i,j}$ for a single path segment within the threat zone is graphically presented in Figure 25. The linear cost function used previously is also presented for comparison purposes. Near the threat, the cost falls off steeply, while the gradient becomes more gradual as the UAV location approaches the outer edge of the threat zone. The non-linear formulation should model the reality of the situation more accurately than the linear formulation. For example, if the UAV is flying close to the SAM site, the missiles have a much shorter distance to fly, there are likely to be fewer obstacles (like mountains) obscuring the UAV’s position, and the UAV can be tracked more precisely. These factors make being close to the threat very dangerous. As the UAV moves father away from the SAM site, the factors combine so that the threat to the vehicle drops off quickly with distance initially. When the UAV is near the edge of the threat zone, the missiles will be traveling much farther and, most likely, there will be more obstacles present, making the likelihood of the missile hitting the UAV much lower. Range makes a difference in relative accuracy: a separation distance of 20,000 feet will not really make the UAV much safer than 19,000 feet while the difference between 6,000 feet and 5,000 feet will be much more pronounced.

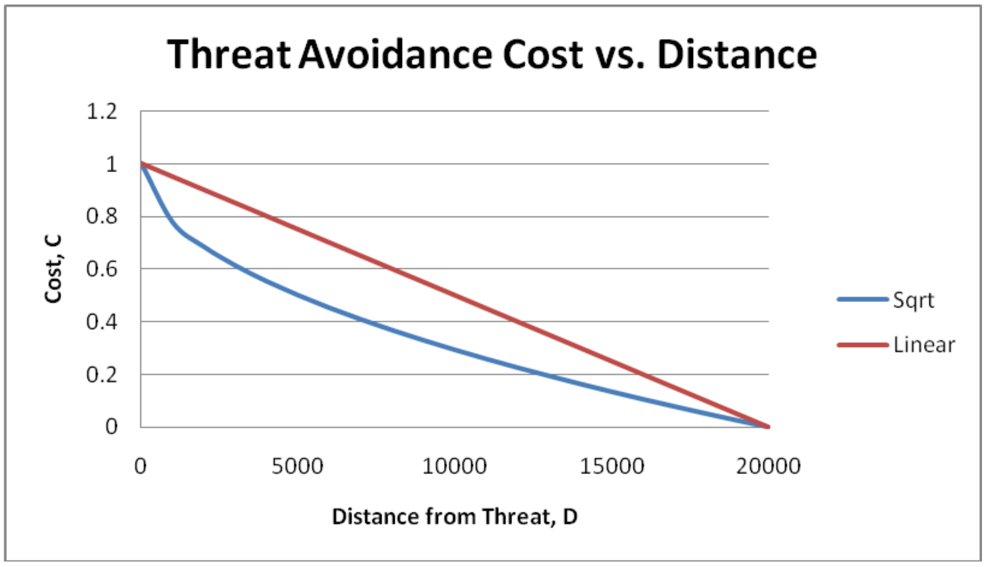


Figure 25. Graph of threat avoidance cost versus distance from threat for a path segment

It is possible for $C_{i,j}$ to be zero for all path segments if the UAV avoids all threat zones completely. If this situation occurs, the 10 that is added will keep the cost value in the proper range.

Terrain Collision Avoidance Component Cost

The calculation for terrain collision avoidance was a completely new addition to the cost function, as seen in Equation 24. The goal of this component was to prevent infeasible paths from being returned to the operator. Unlike the other components, T has no scaling factor, allowing the objective to be scaled without affecting the terrain avoidance portion of the algorithm. This formulation more accurately models the desires of the operator: avoiding the terrain is necessary to generating feasible paths regardless of how the other objectives are weighted.

$$C = K_1 C_F + K_2 C_R + K_3 C_T + T \qquad \textbf{(24)}$$

During the optimization process, the PSO algorithm uses the terrain height information to check if the alternate paths are feasible and safe. A feasible path does not violate the terrain boundary while a safe path is one that stays a user-specified distance above the terrain. That distance was set to 500 feet for testing purposes but is user adjustable.

Every path segment $p(u_i)$ is checked against the terrain. The terrain height value, $p_{terrain}(z_i)$, corresponding to the segment's x, y location, is given by $[p(x_i), p(y_i)]$. The terrain separation distance h_i is then readily obtained with Equation 25. This is shown graphically in Figure 26, where the red rectangle represents the four terrain points used to interpolate the terrain height and then to find h_i for that path segment.

$$h_i = p(z_i) - p_{terrain}(z_i) \tag{25}$$

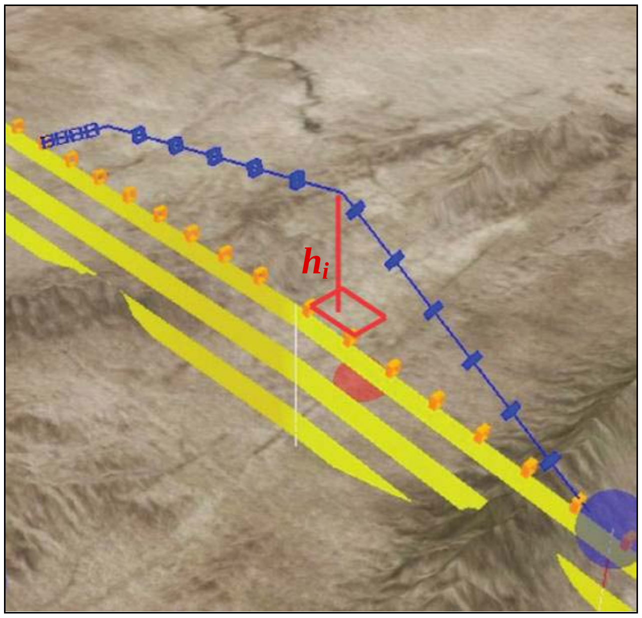


Figure 26. Calculating terrain separation distance

The value of h_i is dealt with according to the logic of Equation 26. The values 1000 and 500 are user-defined parameters and can be adjusted to suit the operator's desires. If the UAV is above the minimum safe altitude, no cost is incurred. If it is lower than that, the cost linearly increases from zero at the minimum safe altitude to 10 at the actual terrain level. If there is a terrain violation (h_i is negative), the cost is a large user-defined value for that path segment. The terrain costs T_i are then summed over all path segments and added to threat avoidance cost component, as seen in Equation 35.

$$T = \sum_{i=1}^{N} T_i \ , \qquad T_i = \begin{cases} 1000, & h_i \le 0 \\ \dfrac{500 - h_i}{50}, & 0 < h_i \le 500 \\ 0, & h_i > 500 \end{cases} \tag{26}$$

Handling terrain avoidance in this manner (as a cost rather than as a constraint) does not guarantee path feasibility. There are two factors motivating this type of formulation. First, it is often the case in optimization that good solutions will lie on the boundary of the feasible design space (good paths may be ones that closely follow the terrain). To find a good solution, the path will often move into infeasible territory (violate the terrain at some location) and then get pushed back again into feasible territory. This exploratory behavior would be hindered by treating terrain as a constraint, which would require either re-initializing the path or altering the curve control points. Second, violating the terrain automatically increases the fitness value for the infeasible path to roughly 10 times that of a feasible path, making it very unlikely that the path would be presented to the operator.

Path Presentation and Selection

The first half of this chapter has focused on Particle Swarm Optimization and cost function development to find optimal solutions to the path planning problem. The second half of the chapter will focus on the techniques developed to present this path information to the operator for the purpose of selecting a suitable alternate path. While much attention in literature has been focused on generating alternate paths, the methods for operator involvement incorporated in the architecture of this path planner are novel. The motivation for developing these interfaces is the need to maintain a human-in-the-loop and facilitate supervisory control of the actions of the UAV. If the operator is going to be able to exercise control over (or make meaningful decisions about) the course of the UAV, that person needs options from which to choose.

Meta-path Display Technique

Usually, only the best solution found by an optimization algorithm is preserved for further evaluation or implementation. This is the premise held by the original path planner formulation. PSO does not limit the operator to just this one solution. As an evolutionary algorithm with a swarm of particles exploring the design space (each one being a potential alternate path), PSO could return multiple alternate paths to the operator for evaluation. In a multi-modal problem, some of these paths will represent other local minima in the design space that might be good solutions to the problem.

Presenting more paths to the operator increases the likelihood that a suitable alternative will be found. Conversely, if too many paths are presented, the operator could be overloaded with visual information and not be able to make a good decision. From testing of the display technique, it was found that using five of the top paths from each of the three runs of PSO was a good balance.

If the algorithm produces five alternate paths and is run with three different weightings, this would produce 15 alternatives. Presented at once, 15 paths is too much information for an operator to evaluate, as is apparent in Figure 27. In order to avoid overloading the operator with this visual information, the alternate paths are grouped into three meta-paths (one for each PSO run). The meta-path representation of the same information is given in Figure 28. With meta-paths, the display is designed to be less cluttered and easier to understand.

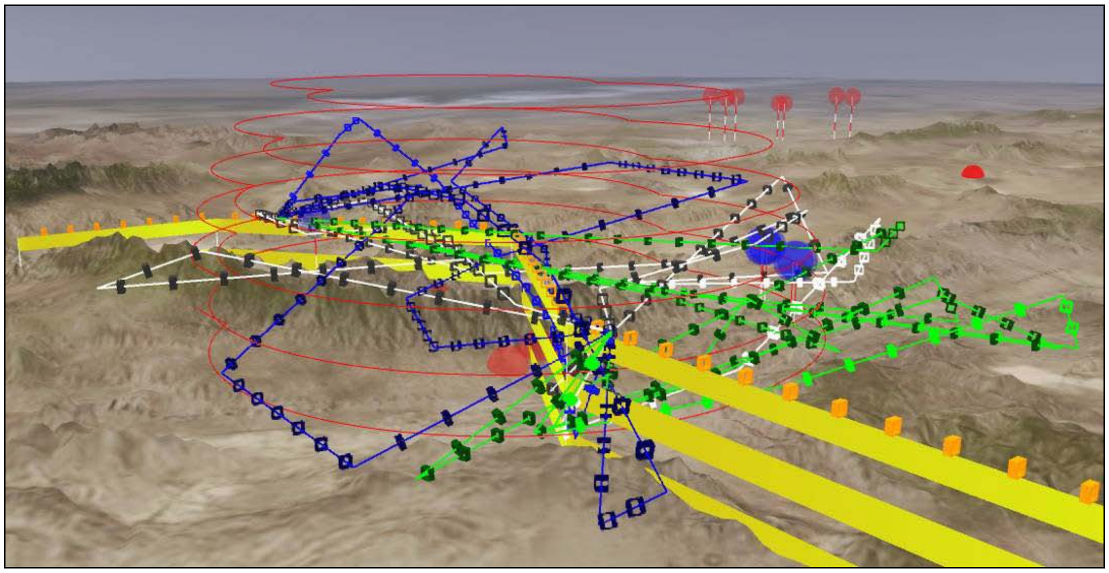


Figure 27. Fifteen alternate paths displayed at one time

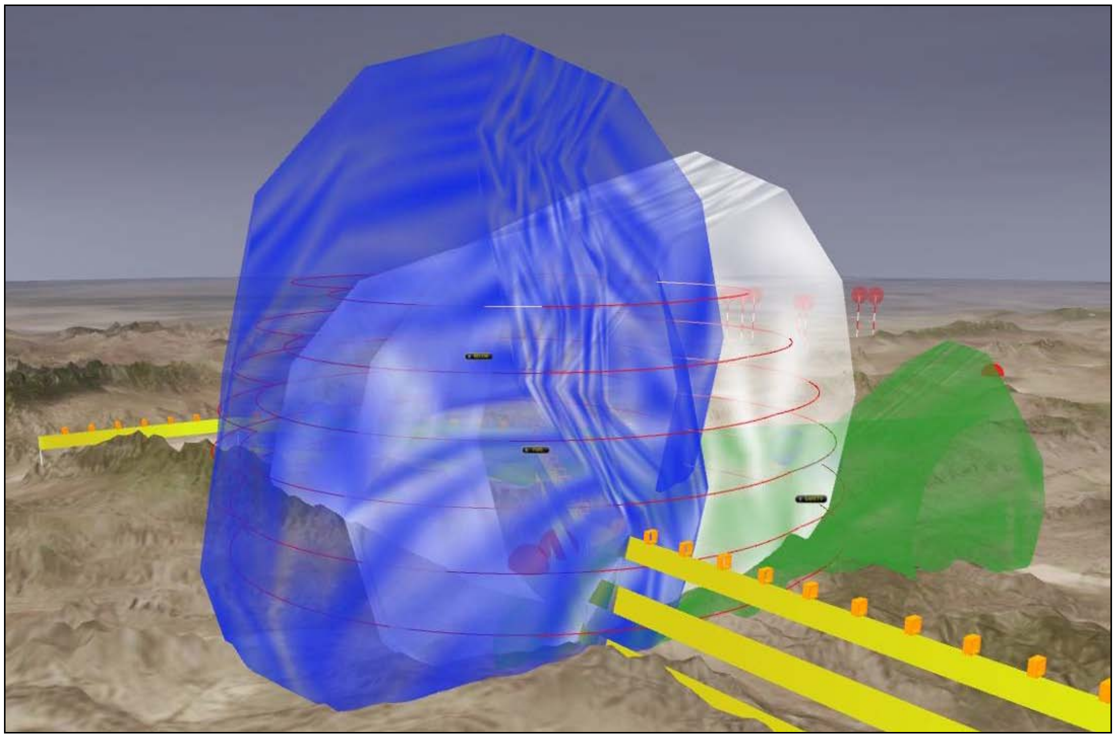


Figure 28. Fifteen paths condensed into three meta-paths

The meta-path representation forms an intermediate step in the decision-making process. The operator now begins by selecting a meta-path. That meta-path is then replaced by the specific alternate paths (default of five) from which it was made. At this stage, the operator can select one of the options and confirm it to update the UAV path. If none of those solutions are desirable, the operator can revert to the meta-path level and select another set of paths to view. The path planning process terminates whenever the operator confirms a path selection or quits the process. By adding an abstracted tier of information and one extra decision, the operator is able to investigate many alternate paths more effectively.

Meta-Path Creation

The first step in the process is to collect the swarm members from PSO when it converges to a solution. The goal is to extract the top unique paths from the swarm, so the members are ordered by fitness from lowest to highest, with lower fitness values indicating more optimal paths. Ideally, the top unique paths would be the first in the list. Practically, swarm members tend to become grouped in the optimal locations of the design space so it is likely that there are duplicate paths (swarm members) in the top of the list, which is why it is necessary to check for similarity among paths and not select paths based on fitness value alone.

Each potential path is compared with the paths that have already been selected for display. Equation 27 and Equation 28 give the similarity calculations. Both paths are broken into line segments. The square of the distance, in 3D space, between the corresponding segments is found and summed along the entire path length. If the value is larger than a user-defined constant C_S, the paths are considered physically distinguishable and unique. This comparison is completed for each of the *n* paths that have been selected and if the current path fails any comparison, it is rejected. If it is physically unique relative to all the selected paths, it is then added to the list.

$$S_n = \sum_{i=1}^{N} \left(p_{cur}\left(x_i\right) - p_n\left(x_i\right) \right)^2 + \left(p_{cur}\left(y_i\right) - p_n\left(y_i\right) \right)^2 + \left(p_{cur}\left(z_i\right) - p_n\left(z_i\right) \right)^2 \tag{27}$$

$$valid = \begin{cases} true, & S_n \geq C_S \text{ for all n} \\ false, & S_n < C_S \text{ for any n} \end{cases} \tag{28}$$

If the desired number of unique and feasible paths are not found, it is possible that infeasible paths will be returned to the operator. There are also cases where there are not enough unique solutions (including infeasible solutions) and then the algorithm returns multiple identical solutions with the duplicates being the last path investigated (the worst path). For example, if there are only three unique paths, the operator will see only those three paths, but the third path will be presented multiple times. However, the user will only see and be able to select three paths.

Once the alternate paths are selected, they are used to generate the meta-path that will be displayed to the operator. This single visual representation conveys information about the location and variability of the alternate paths. A meta-path is a bent, variable-diameter tube. It is textured with transparency so that occluded meta-paths are still visible and is colored in the same manner as the alternate paths that compose it. The two variables defining the shape of a meta-path are the locations of the bend points and the tube radius at each bend.

The *j* bend locations, $p_{meta,j}$, are determined by the locations of the *j* waypoints of the *i* alternate paths, $p_{i,j}$, as shown in Equation 29. Each corresponding waypoint on the alternate paths is averaged in 3D space to generate the meta-path bend location, so there are as many bends in the meta-path as there are waypoints in each alternate path. The radius of the tube at each point, $radius_j$, is designed to represent the spread of the corresponding underlying waypoints of the alternate paths. Waypoints that are far apart will result in a large tube diameter and waypoints that are close together will result in a small diameter. Equation 30 shows how the distances from the meta-path bend point $[p_{meta}(x_j), p_{meta}(y_j), p_{meta}(z_j)]$ to each of the five waypoints $[p_i(x_j), p_i(y_j), p_i(z_j)]$ used to compose it are averaged to determine the tube radius.

$$p_{meta-path}\left(u_j\right)=\frac{1}{5}\sum_{i=1}^{5}p_i\left(u_j\right) \tag{29}$$

$$radius_j=\frac{1}{5}\sum_{i=1}^{5}\sqrt{\left(p_{meta}\left(x_j\right)-p_i\left(x_j\right)\right)^2+\left(p_{meta}\left(y_j\right)-p_i\left(y_j\right)\right)^2+\left(p_{meta}\left(z_j\right)-p_i\left(z_j\right)\right)^2} \tag{30}$$

Since the meta-path parameters are calculated from the underlying paths that compose it, the representation should be, visually, a cloud generally encompassing the alternate paths and should provide the operator with clues as to what the individual paths are doing without actually showing the paths.

Augmented Interface for Path Evaluation

After selecting a meta-path, the operator must choose one of the alternate paths. Because the model used in optimization does not perfectly match the real world situation or the operator's preferences, the algorithm's optimal choice may not be the best alternate path. The operator needs some tools to differentiate between the alternatives besides their visual representation. Only one of the alternate paths is mathematically optimal, according to the criteria given to the PSO algorithm. It would be helpful for the operator to know how far from optimal the other paths are as that may impact the decision that is made. For example, if the operator has one choice that looks slightly better than another but it is 30% less optimal, the more optimal choice might actually be selected.

Relative fitness was created to provide a gauge of optimality. It is computed as the ratio of the fitness of the optimal path ($fitness_{path\ 1}$) over the fitness of the current path ($fitness_{path\ i}$) as in Equation 31. The calculation is performed for each of the alternate paths, *i*, generated by one run of PSO. Thus, each meta-path will contain an optimal alternate path. In the display, the current path is always highlighted and its fitness is presented as a number between zero and 100, with 100 being the optimal path. If the relative fitness is 95 percent, the operator can think of that path being five percent less than the optimal path. It is important to note that these relative fitness values only hold for the given fuel efficiency, reconnaissance, and threat avoidance weights used in the optimization.

$$relative\ fitness_i = \frac{fitness_{path\ 1}}{fitness_{path\ i}} \times 100\% \qquad \textbf{(31)}$$

The operator might also be interested in how alternate paths rank with one another in terms of individual objectives. For instance, the operator may be most concerned about threat avoidance, but also interested in fuel efficiency to make sure the UAV returns to base undamaged. In this case, the operator might select the threat avoidance meta-path and then want to see how the alternate paths rank in terms of fuel efficiency.

A set of three meter-bars, similar to what would be seen in a video game for rating character abilities, are used to present the relative component fitness values for the currently selected path. The calculations for fuel efficiency, reconnaissance, and threat avoidance are given in Equations 32 through 34. For fuel efficiency, the best fuel component cost among all alternate paths is found and set as $\textit{fuel fitness}_{best}$. The fuel fitness for a specific path ($\textit{fuel fitness}_i$) can then be found by dividing the best value by the fuel component cost for that path ($\textit{fuel fitness}_{path\ i}$). The calculations are identical in the other two cases, just substituting out for the appropriate objectives. The component fitness costs are compared apart from their weights, so it is valid to compare cost values between meta-paths. This is done to give the operator an understanding of how the highlighted path ranks against all the other alternatives. The component fitness value determines the fraction of the bar that is full for that objective. So, if for a set of five paths, all the bars are low for one objective, there is at least one (and most likely more than one) alternate path in one of the other groups that more effectively minimizes it. If that objective is important to the operator, it might lead to the decision to investigate another meta-path that does better in that area.

$$\textit{fuel fitness}_i = \frac{\textit{fuel fitness}_{best}}{\textit{fuel fitness}_{path\ i}} \tag{32}$$

$$\textit{recon fitness}_i = \frac{\textit{recon fitness}_{best}}{\textit{recon fitness}_{path\ i}} \tag{33}$$

$$threat\ fitness_i = \frac{threat\ fitness_{best}}{threat\ fitness_{path\ i}} \quad \textbf{(34)}$$

A screenshot of the Virtual Battlespace showing these two features for path evaluation is given as Figure 29. In this screenshot, the highlighted path (drawn over with a thicker blue line) has a relative fitness of 100, meaning it is the optimal reconnaissance path. In the figure the path shown is good, but not optimal, in all the component areas. Thus, the path can be optimal overall without being the best in any single category.

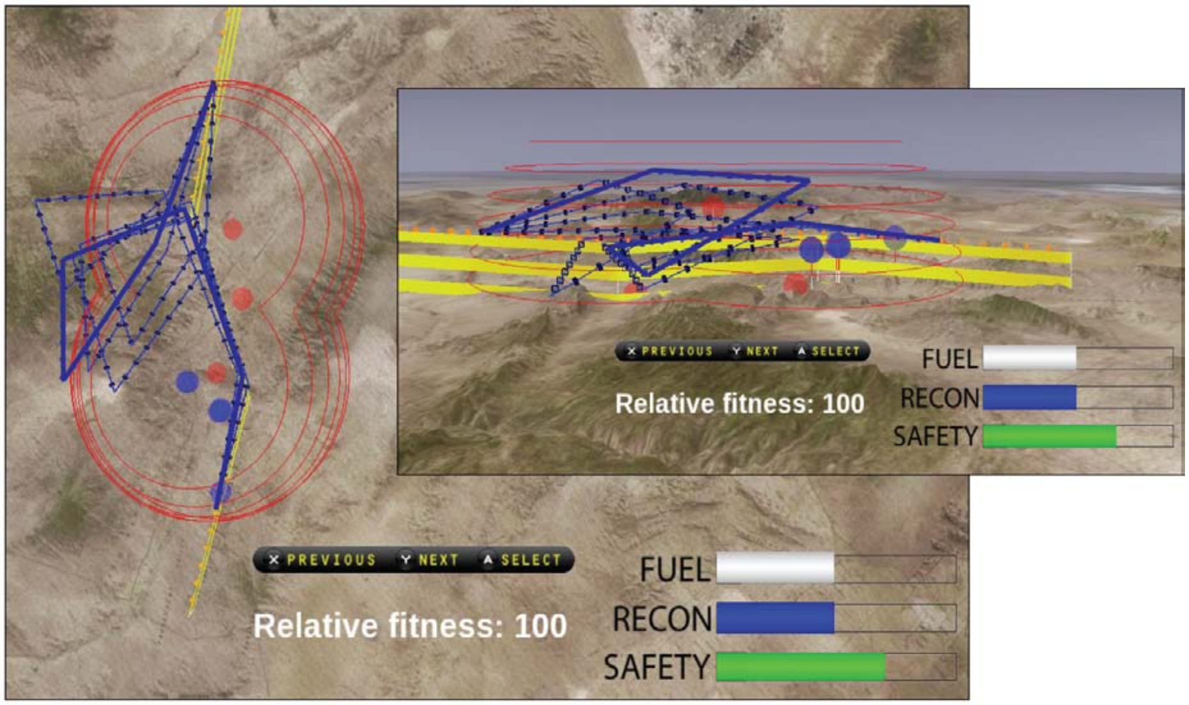


Figure 29. Virtual Battlespace in path planner mode with relative fitness and objective fitness bars displayed

User-Interaction in Path Planning

Meta-paths do not address the issue of more fully incorporating user preferences into the optimization algorithm. The weights for the component costs are set before the code is run and cannot be modified. Two methods were explored to allow the operator to interact with the weights and change the paths that are displayed in real time.

Interactive Weight Adjustment

The first method allows the operator to adjust the component weights and re-run the PSO algorithm. In this framework, when a path planning alert comes up, the operator is given the ability to set the weights for the algorithm and then tell it to optimize (using those weights). The weights are adjusted using the bars in Figure 30.

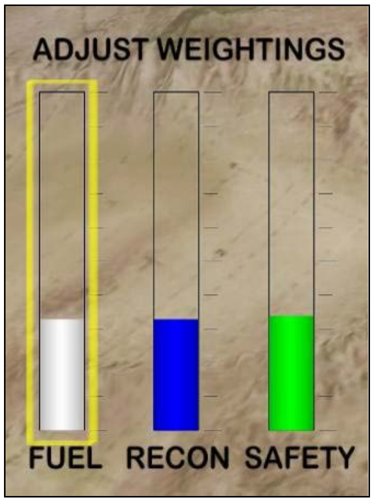


Figure 30. Adjustable weight bars

To allow this type of interaction, the PSO algorithm was modified slightly to operate more quickly. Instead of performing three separate optimization runs, only one is done. Since the operator provides the weights to the algorithm, no guesswork is necessary, eliminating the need for the separate optimizations. With these changes, the algorithm is able to re-plan a path based on the user input (how the weights are set) and display that path in real time.

Post-Planning Weight Adjustment

The second interaction method for decision-making is a post-analysis of the paths generated by the PSO algorithm. The method is presented as a flowchart in Figure 31.

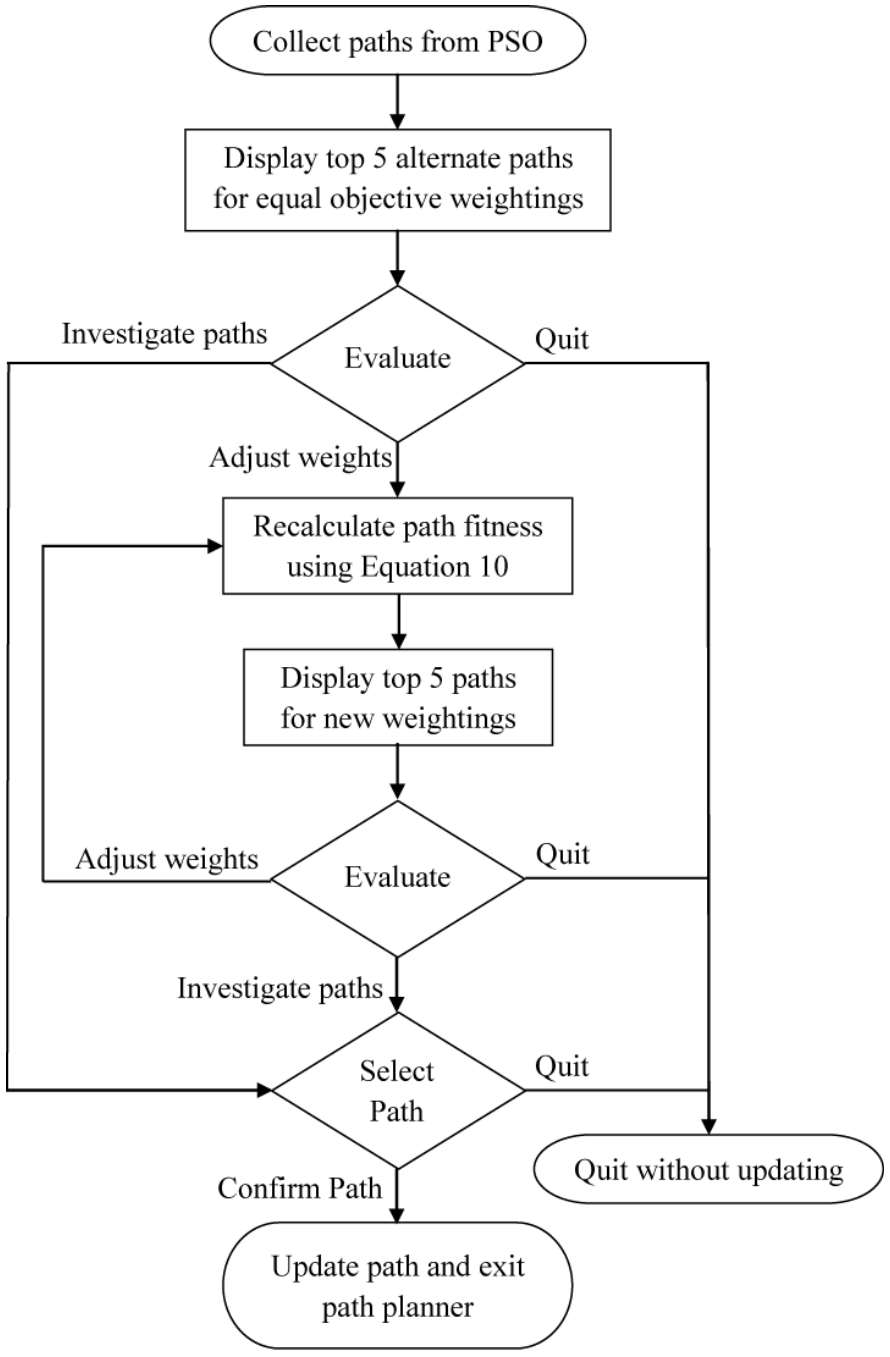

Collect paths from PSO
Display top 5 alternate paths for equal objective weightings
Investigate paths
Evaluate
Quit
Adjust weights
Recalculate path fitness using Equation 10
Display top 5 paths for new weightings
Adjust weights
Evaluate
Quit
Investigate paths
Select Path
Quit
Quit without updating
Confirm Path
Update path and exit path planner

Figure 31. Flowchart for post-analysis of PSO data for decision-making

PSO operates as in the other cases, optimizing once each for fuel efficiency, reconnaissance, and threat avoidance. The weights used in optimization are not adjusted by the operator. Following optimization, instead of saving just a user-defined number of paths, all unique paths (with their un-weighted component costs) are stored.

When the PSO algorithm finishes optimizing, a set of three adjustable bars (one for each objective) is presented to the operator. Initially, the bars are set so all three objectives are equally weighted. The operator adjusts these bars as desired. As the weights (K_1, K_2, and K_3) change, the fitness costs for the stored paths are updated (according to Equation 10) in real time. The top paths, according to those weights, are then displayed to the operator. The operator can continue to adjust the weights until a desired set of paths is presented.

Then, the operator can choose to leave the weighting mode and investigate the alternate paths displayed on the screen. In this mode, interaction is similar to viewing alternate paths after selecting a meta-path. The current path is selected and its relative component fitness bars are displayed. The operator can select any one of the alternate paths to replace the original path and re-task the UAV. As in the other cases, the operator also has the option to exit the path planner at any point without re-tasking the UAV.

Chapter 4 – Results and Discussion

The changes to the cost function as well as the improvements to the interface presented in the method development chapter were analyzed and evaluated to determine their effectiveness in aiding the operator in re-tasking the UAV. The analysis moves from the low-level mathematical formulations of the cost functions to the high-level interaction between the operator and the path planner. Both quantitative and qualitative measures were used to determine if these goals were met.

Evaluation begins with a comparison of the original and final cost functions implemented in PSO. The cost function formulation impacts both solution time and the characteristics of the alternate paths. Each cost function will be evaluated based on its performance on four test scenarios. Using multiple distinct scenarios provides a more complete understanding of the characteristics of the cost functions.

Test Scenarios for Evaluating Cost Function Formulations

Each test scenario will be presented in turn, from the simplest to the most complex. The first test scenario, shown in Figure 32, is the simplest case possible. One ground threat (the red entity on the terrain) is positioned close to the UAV’s original path (the yellow fence). There is only one waypoint within range of the threat (the first slight bend in the path). The UAV has many options for attempting to avoid the threat while staying close to the waypoint and not expending fuel unnecessarily.

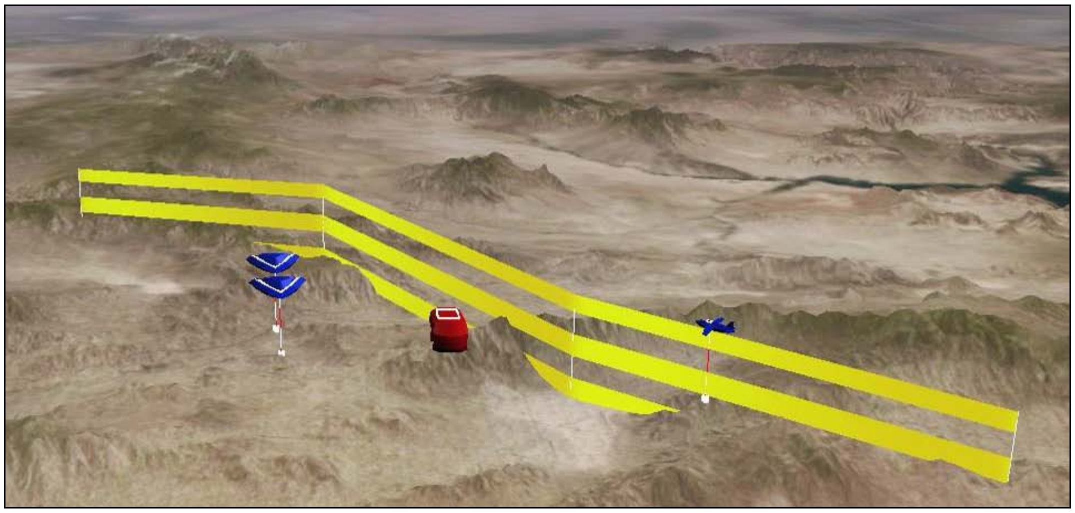


Figure 32. Test scenario one with one ground threat

The second test scenario, shown in Figure 33, contains one aerial threat (the red airplane) that the UAV must avoid. This situation should differentiate between the cost functions that include terrain collision avoidance and those that do not. The aerial threat is at a higher altitude than the UAV, making it more efficient, in terms of path length, for the UAV to dive under the threat than try to go over it. Unless terrain is considered, the alternate paths are likely to dive too far and go through the terrain to avoid the threat.

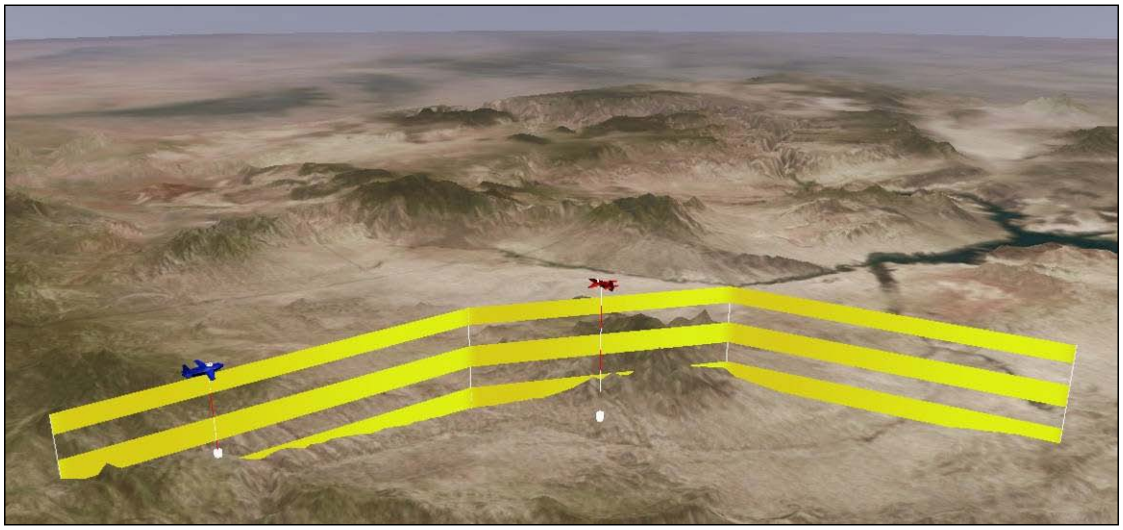


Figure 33. Test scenario two with one aerial threat

Test scenario three creates a more complicated situation. Instead of one threat, there are now two entities the UAV must avoid; one on the ground and one in the air. Figure 34 presents a top view and side view of this scenario. The fourth test scenario is the most complex situation. It incorporates three distinct threats; two on the ground and one in the air. The threats lie on both sides of the original path and interfere with the UAV's ability to safely pass over the two middle waypoints (reconnaissance points) of the original path. Both of these cases are designed to see how the cost function formulations perform in more complicated situations where the optimal path is less readily identified. The top view and side view of test scenario four are shown in Figure 35.

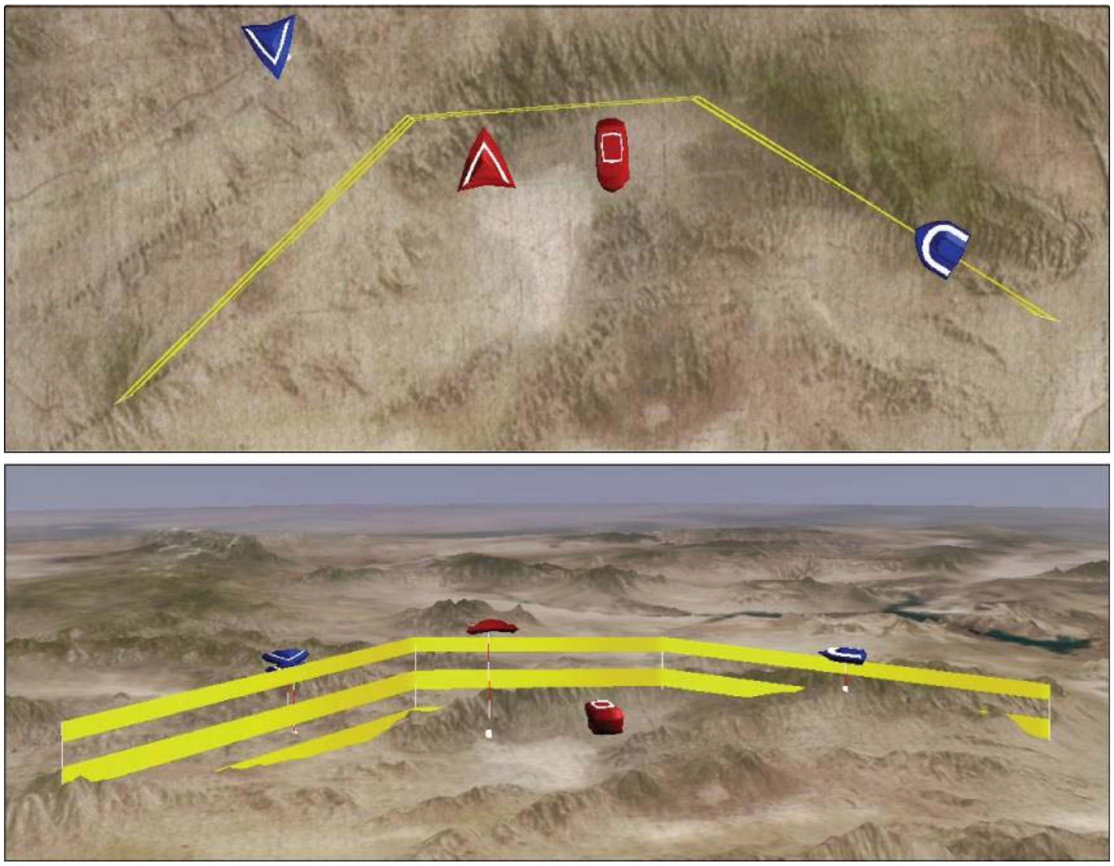


Figure 34. Test scenario three (top view and side view) with a ground threat and an aerial threat

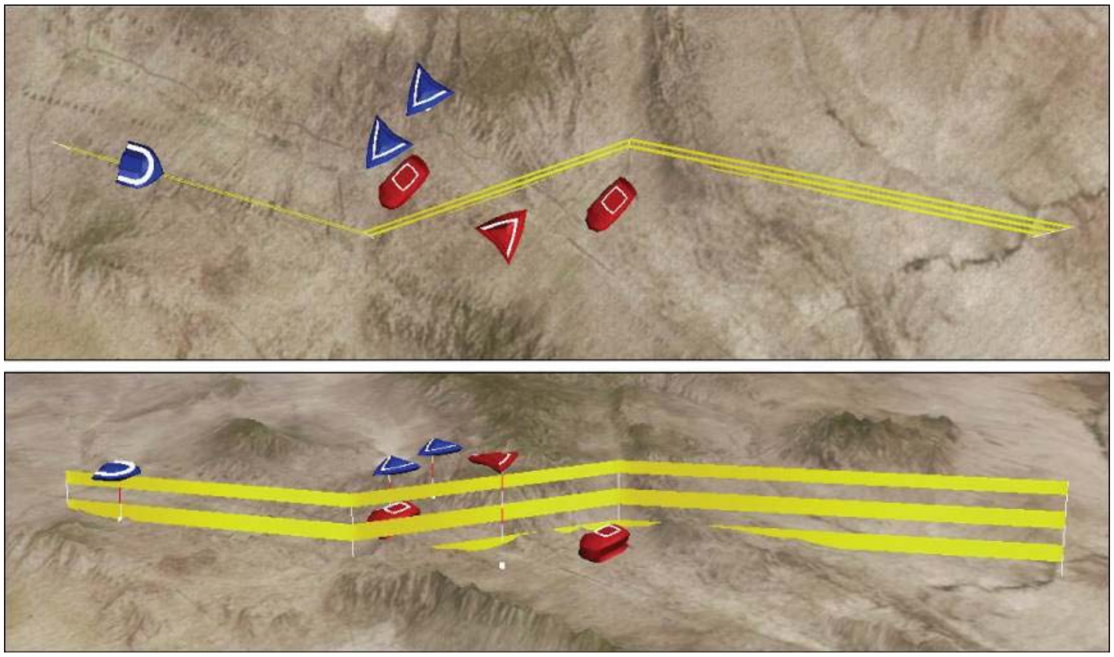


Figure 35. Test scenario four (top view and side view) with three threats

Cost Function Formulation Analysis

The four scenarios (summarized in Table 2) were used to test the original and final formulations. Each test case was run four times and the results were averaged over the four runs. PSO is an evolutionary algorithm which uses a randomly initialized swarm of particles to explore the design space and find the optimal solution. For that reason, PSO will not solve the problem exactly the same way twice: sometimes it may find an excellent solution almost immediately, while in other cases, it may require a different number of iterations to find an optimal path. Thus, the PSO algorithm was run multiple times on the same problem to create an average result. Quantitative and qualitative measures were used to evaluate the cost functions.

Table 2. Description of scenarios used in testing

Scenario	Description
1	One ground threat (simplest possible case)
2	One aerial threat
3	One ground and one aerial threat
4	Two ground and one aerial threat with distinct threat zones

Calculation time was the first measure of interest. The total time the operator has to re-task the UAV (30 seconds) includes the time required by PSO to generate the alternate paths. The less time the algorithm takes calculating solutions, the longer the operator has to evaluate the alternatives and implement one of them. Figure 36 presents a graph of the calculation times (in seconds) for the cost functions for each scenario. The last bars on the right are the averages of the other four bars.

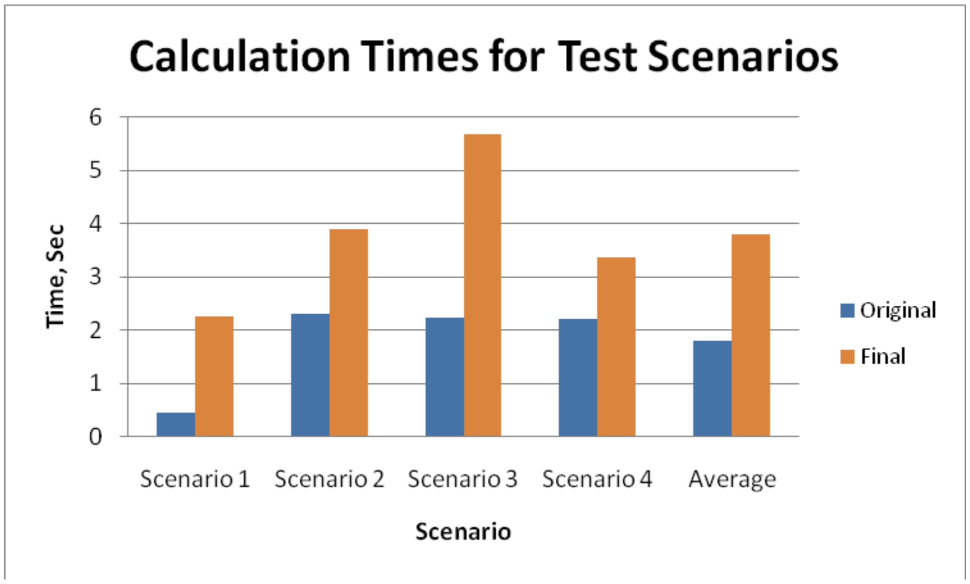


Figure 36. Graph of calculation times for the cost functions for each test scenario

As expected, the original formulation requires less time to solve the path planning optimization problem as it is a simpler set of calculations and does not account for terrain avoidance. The final formulation takes on average two seconds longer than the original formulation to solve the same problems. The cost (in time) of this added complexity needs to be outweighed by the benefits to the operator (in better alternate path choices) for the final formulation to be considered better than the original.

The performance of the cost functions was also evaluated based on the number of iterations to convergence. When planning a path, PSO performs three separate optimizations on the same scenario, one for fuel efficiency (fuel), one for reconnaissance (recon), and one for threat avoidance (threat). The difference between the optimization runs is the weights assigned to the objectives. Table 3 presents the weight coefficients used in these tests.

Table 3. Component weights for the objective function for the three PSO optimization runs

	Fuel Weight, K_1	Recon Weight, K_2	Threat Weight, K_3
Fuel efficiency path	0.60	0.20	0.20
Reconnaissance Path	0.20	0.60	0.20
Threat Avoidance Path	0.20	0.20	0.60

When the iterations from all three optimizations are grouped, the original formulation takes 207 iterations to complete while the final formulation takes only 177. This shows that the increase in calculation time is not due to PSO needing more iterations to find the optimal path, but rather computational complexity retrieving and analyzing terrain data.

Figure 37 shows the breakdown of the optimization iterations for each objective. For each set of three bars, the only parameters that change are weights attached to the components. It is notable that test scenario one takes half as many iterations to solve as the other cases. This would fit with scenario one also taking half as much time to optimize. The only significant difference between it and the other scenarios is that only one reconnaissance point is affected while two are affected in the other cases. This suggests that the number of reconnaissance points has a significant impact on solution time and would be a point worth further investigation. One other anomaly is the iteration count for the fuel efficiency paths for the final cost function for scenario four. Upon a more detailed investigation, it was found that some of the runs converged very quickly, causing the average to be very low. In general, the similarity between the original and final algorithm in this area suggest that PSO is functioning fundamentally the same in both cases.

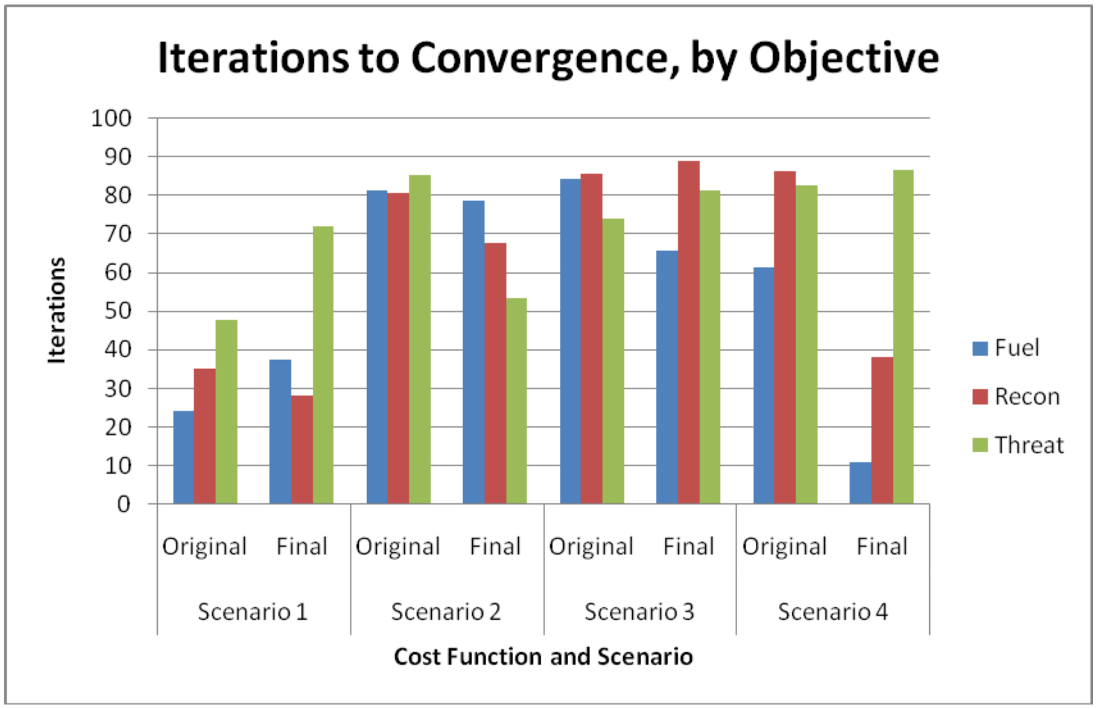


Figure 37. Iterations to convergence for PSO optimizations weighted to favor fuel efficiency, reconnaissance, and threat avoidance

A final numerical analysis of interest is comparing the cost function values for each of the objectives. As mentioned in the method development section, having component cost values that are separated by orders of magnitude would make the problem more difficult for PSO to optimize in the manner the operator intends. Figure 38 presents the component cost values for the original cost function. The graph can be thought of as a decomposition of the objective function where each component value is presented separately. In order, the four bars for fuel paths are the fuel efficiency component cost (in blue), the reconnaissance component cost (in red), the threat avoidance component cost (in green), and the total cost (in purple). The total cost is the objective function value that PSO is trying to minimize.

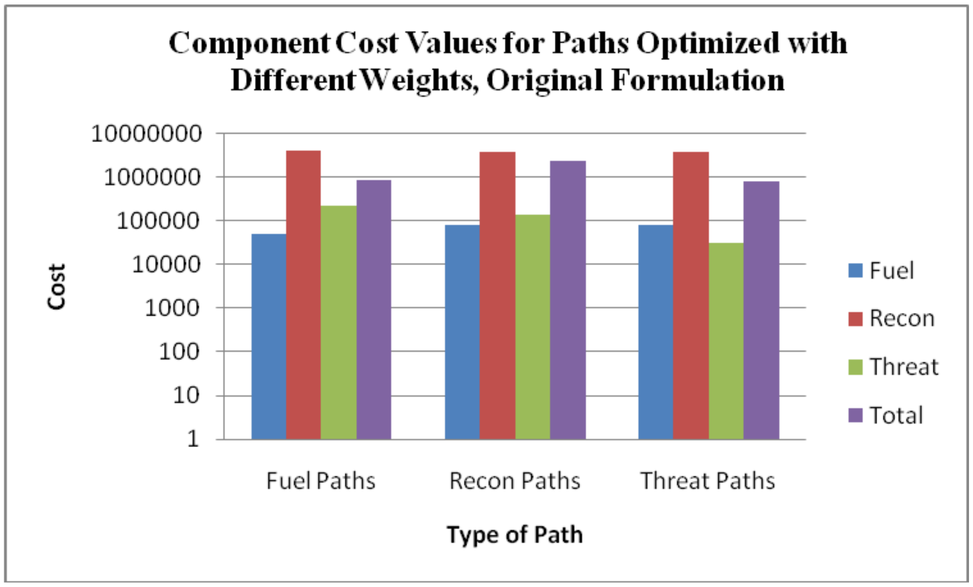


Figure 38. Component cost values for the original cost function

The graph clearly shows the difference in magnitude of the component cost values (note the logarithmic scale for cost values). In all cases, the reconnaissance cost is 100 times as large as the other two costs. This would tend to skew paths toward optimal reconnaissance performance as minimizing that value will most effectively minimize the total cost.

Figure 39 presents the same information as Figure 38 but for the final cost function formulation. Now all the component costs are in the same order of magnitude, which should allow PSO to more efficiently optimize the solution and the weights should correspond more closely to what the operator intends.

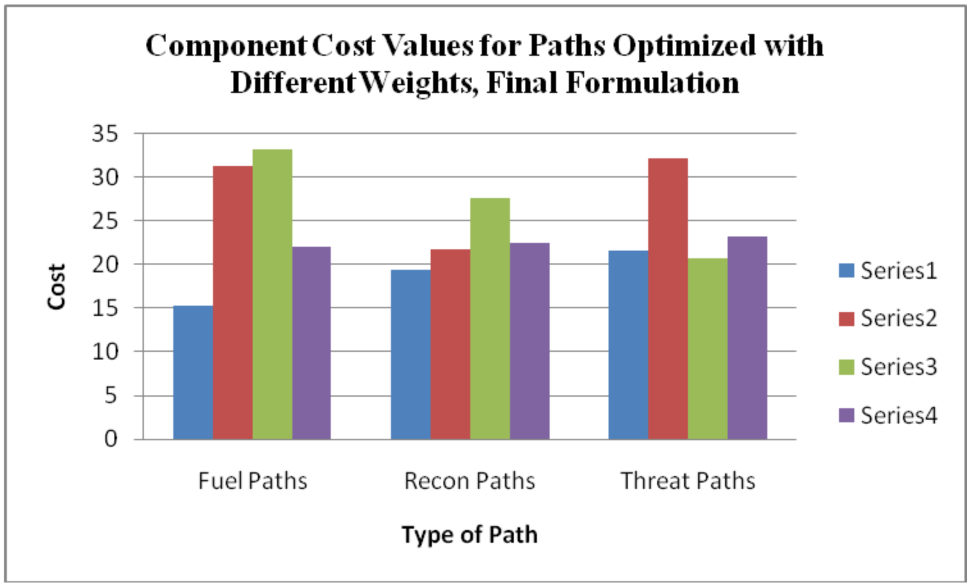


Figure 39. Component cost values for the final cost function

The numerical performance data presented above demonstrate that the improved cost function takes longer to generate solutions. It is also important to analyze the alternate paths visually to understand the types of paths each formulation produces and qualitatively evaluate the improvements to the path planner.

Test scenario two (the case with one aerial threat) will be used to facilitate a subjective comparison of the paths produced by the cost function formulations. For each cost function, the PSO algorithm returns paths optimized for each of the sets of weights listed earlier in Table 3. The fuel efficiency paths are white, the reconnaissance paths are blue, and the threat avoidance paths are green. The alternate paths for the original cost formulation are presented in Figure 40.

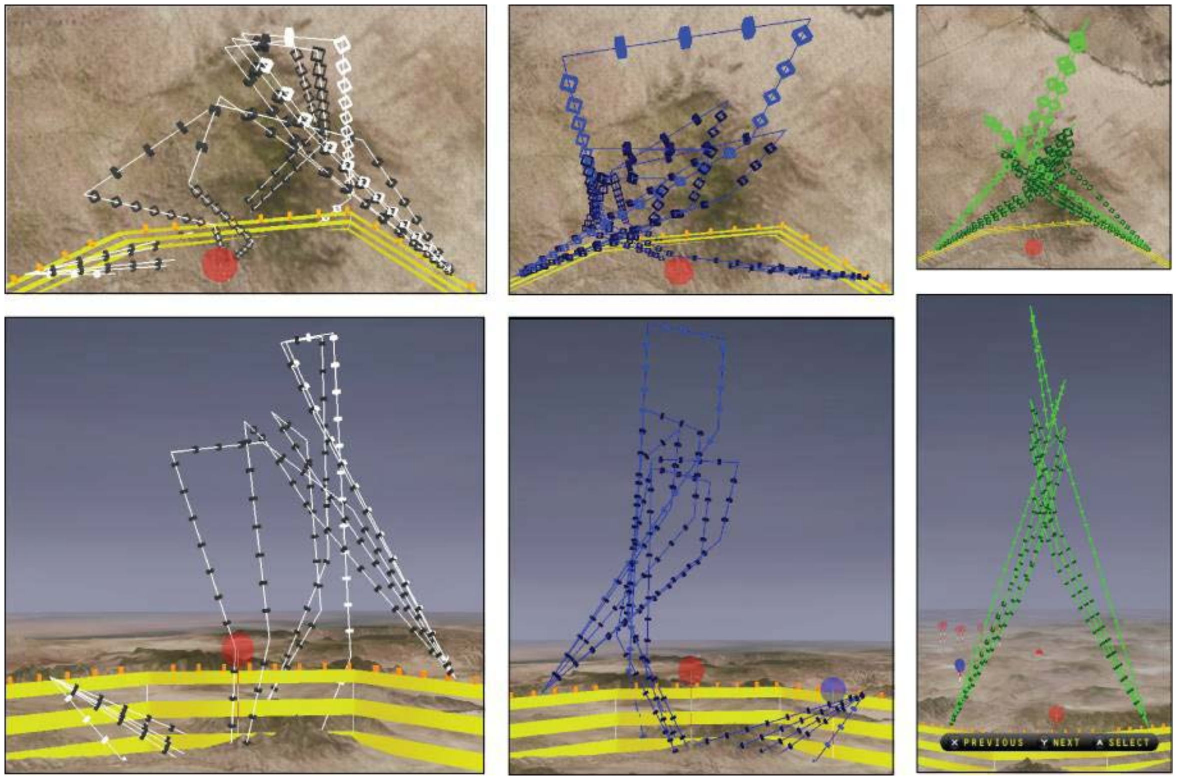


Figure 40. Alternate paths for the original cost function for test scenario two: top and side views are given for fuel efficiency (white), reconnaissance (blue), and threat avoidance (green) paths

The paths generated by the algorithm in this case are clearly not desirable. Both the fuel efficiency and reconnaissance paths attempt to go under and then over the threat, first diving into the terrain (collision avoidance is not considered) and then flying almost vertically to get over the threat. It is surprising that the fuel efficiency paths would follow such long trajectories as the path length could easily be cut in half with a more direct route (significantly reducing the cost). The threat avoidance paths seem to be the most logical. All those paths follow an inverted V, attempting to avoid the threat by going over it. From the top-down views, it is clear that though the paths change elevation drastically, they follow the original path closely in the x and y dimensions. This behavior suggests that reconnaissance tends to dominate the optimal alternate paths. It also leads to paths that are very similar which, practically, does not provide the operator with many options.

Figure 41 shows the paths generated by final formulation of the cost function. The alternate paths are generally less averse to being in close proximity to the threat. This would be expected as the cost for a mild violation of the threat zone is lessened from the previous formulation. It seems that the change in threat avoidance cost lends itself to allowing PSO to find a wider range of alternate solutions, which is beneficial. On the other hand, for the operator who places a high importance on the safety of the UAV this formulation may not be ideal and a more conservative formulation may be better.

The fuel efficiency paths (in white) clearly display the multi-modal nature of the problem and how PSO takes advantage of it. The paths split around the threat, one going right of the original path and other going left. This provides the operator with more freedom in the path-selection process as there is a more diverse set of solutions from which to choose. The fuel efficiency paths also demonstrate what happens when there are not enough unique solutions. Only two were found so only two are presented to the operator for selection.

There are two components that would contribute to the behavior of the reconnaissance paths. First, having the loops can allow the UAV to find a better approach and get closer to the reconnaissance points, generating a better cost. Second, a path is broken into a user-specified number of segments for analysis, regardless of how long it is. The proportional amount of time inside the threat zone is what is being accounted for by the cost and not time. Thus, a longer path that has the same amount of path length inside the threat dome will be less expensive. A more heavily weighted fuel efficiency component might shorten the paths and eliminate those loops.

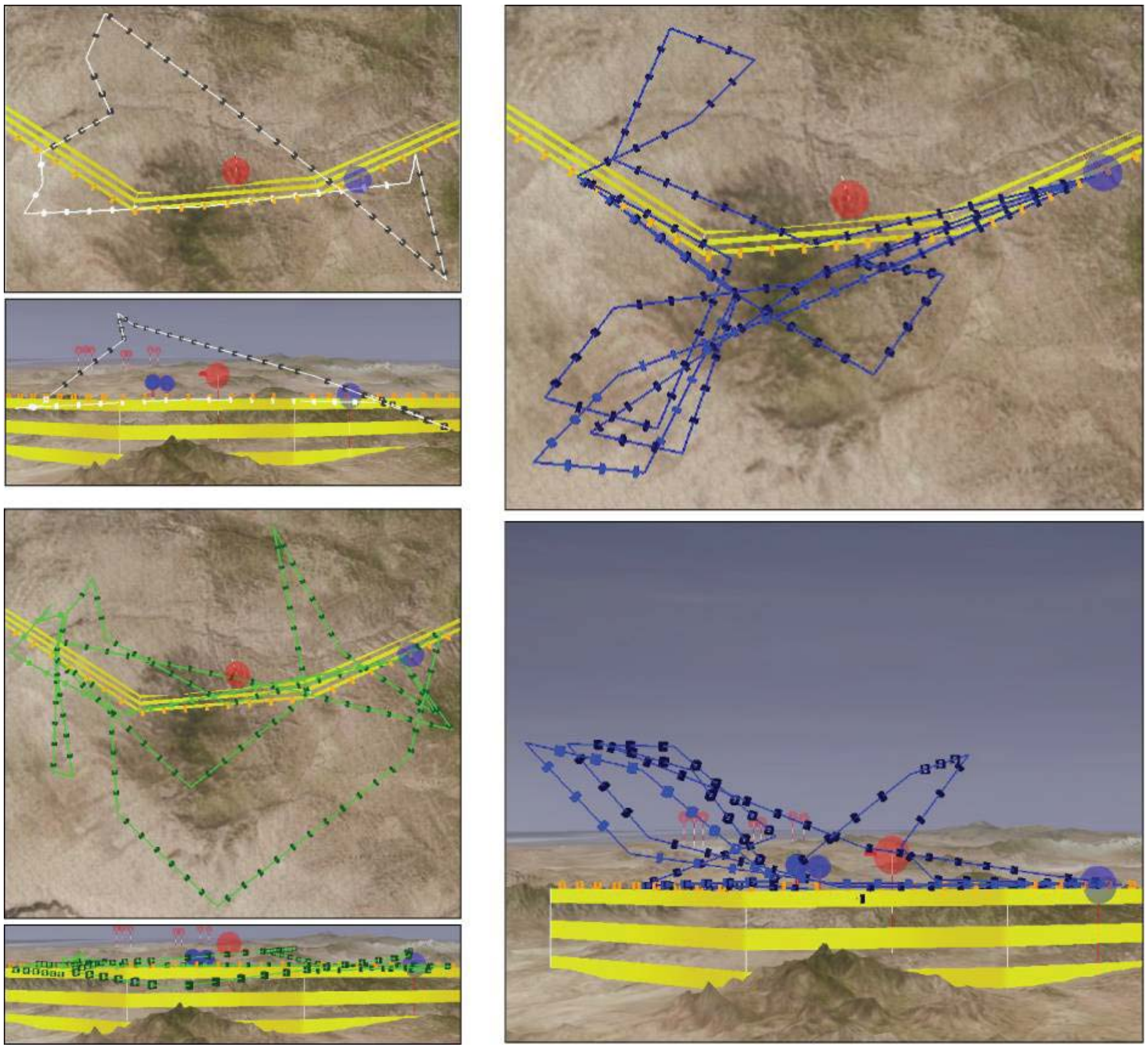


Figure 41. Alternate paths for the improved cost function for test scenario two: top and side views are given for fuel efficiency (white), reconnaissance (blue), and threat avoidance (green) paths

Comparing these paths with those generated by the original cost function demonstrates how much influence the cost function formulation has on the types of paths the algorithm will produce. The extreme behavior and terrain violations from the original cost function is not present in the paths generated by the improved version.

The cost functions were also qualitatively analyzed based on the paths generated for test scenario 4 to see if the trends observed in test scenario two would hold for a different situation. Only the reconnaissance paths are presented here as they seem to be representative of the performance of the cost functions. Figure 42 shows the reconnaissance paths for the original cost function formulation. The paths are all relatively similar and attempt to fly over the threat, which is consistent with what was observed in the previous test scenario. None of these paths would be desirable alternatives due to the near-vertical sections they contain.

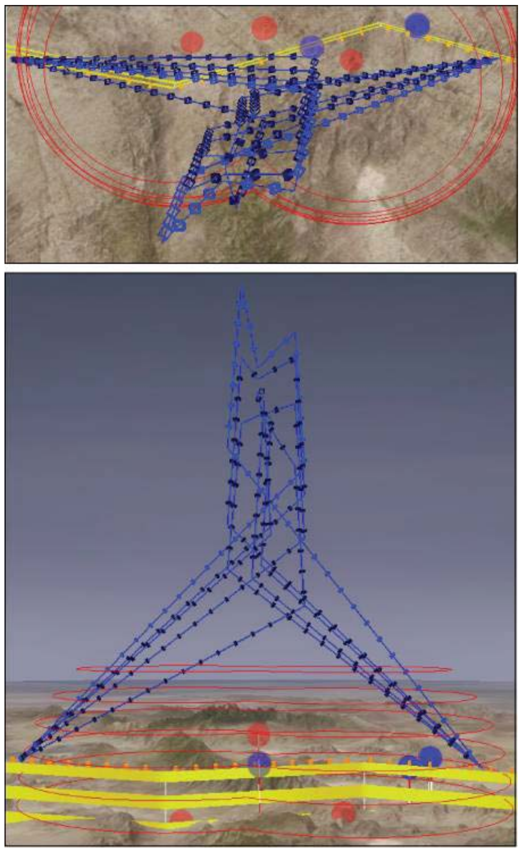


Figure 42. Alternate paths for the original cost function for test scenario four: top and side views are given for the reconnaissance paths

Figure 43 displays the reconnaissance paths for the final cost function. Again, they tend to stay quite close to the original path but this time the loops seen in the previous run are not present. Both cost functions seem to be behaving consistently.

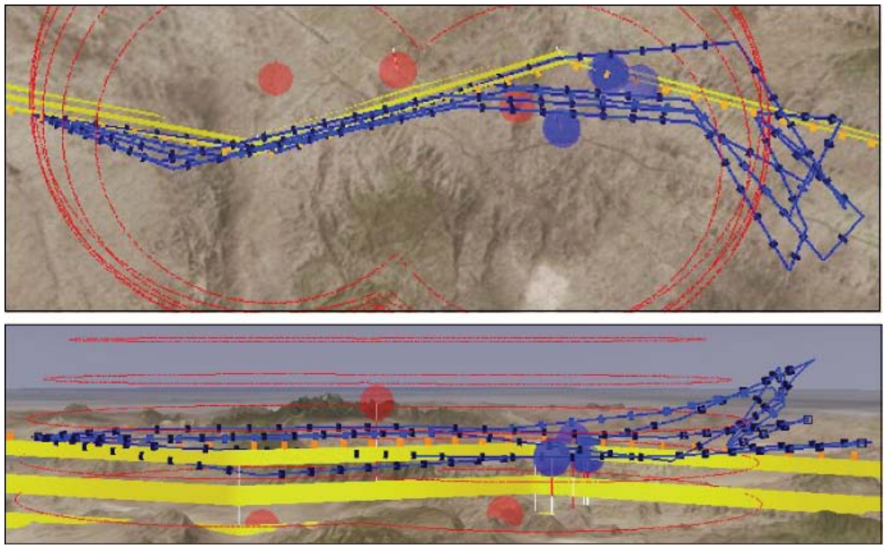


Figure 43. Alternate paths for the improved cost function for test scenario four: top and side views are given for the reconnaissance paths

From the qualitative and quantitative analysis presented in this chapter, the extra time required by the algorithm to compute paths with the new cost functions is well spent in generating more desirable path options. As the terrain component contributes largely to the increase in calculation time, it will be analyzed in more detail to better justify its inclusion in the algorithm.

Impacts of including Terrain Information in the Path Planner

To test the terrain component of the algorithm, six test scenarios (different than those presented above but of similar complexity) were generated. Two scenarios are presented pictorially in detail here. The other four are only included as part of the calculations. The first test scenario is a simple case where there is one enemy fighter plane that is flying close to the UAV's original path, as can be seen in Figure 44.

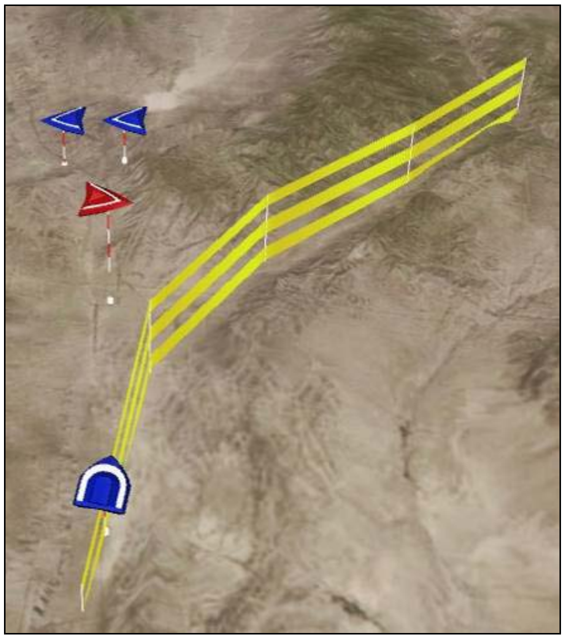
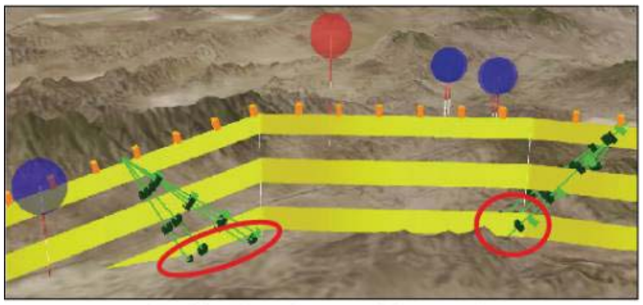
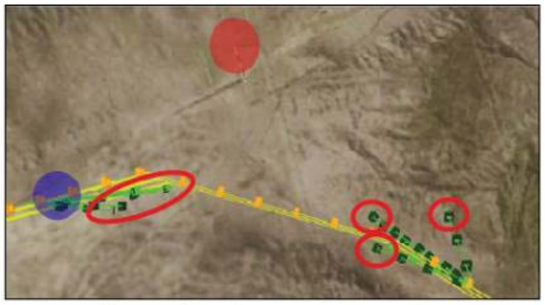
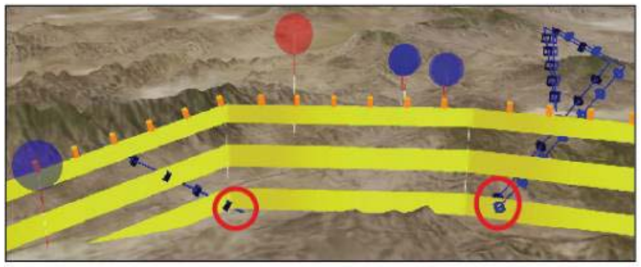


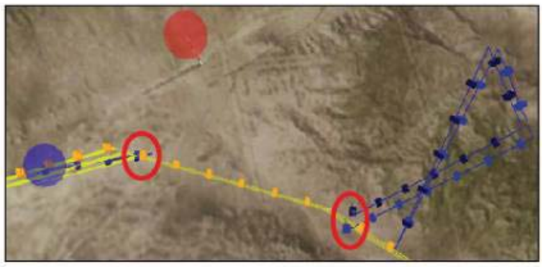
Figure 44. First test scenario with one aerial threat (in red)

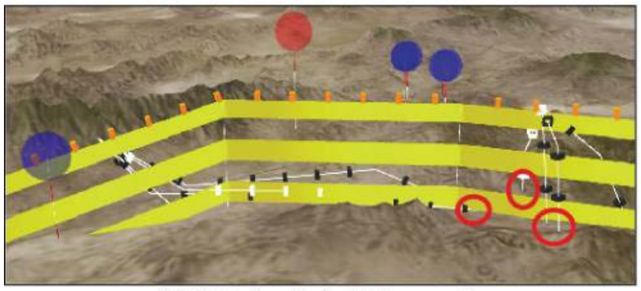
Figure 45 shows the paths that were generated from one run of the PSO path planner algorithm that did not include terrain information. From top to bottom, the paths are weighted toward threat avoidance, reconnaissance, and fuel efficiency. The terrain violations are circled in red in the figure. Since terrain information is not considered the ground does not exist from the viewpoint of the original cost function. To achieve the lowest possible fitness value, the algorithm seeks to minimize path length, which is most readily done by flying under the threat. The algorithm also needs to avoid the threat, which is what causes the paths to dive further, leading to the terrain violations. On this particular run, all 15 of the paths violate the terrain boundary.


a) Right view threat avoidance paths


b) Top view threat avoidance paths


c) Right view reconnaissance paths


d) Top view reconnaissance paths


e) Right view fuel efficiency paths

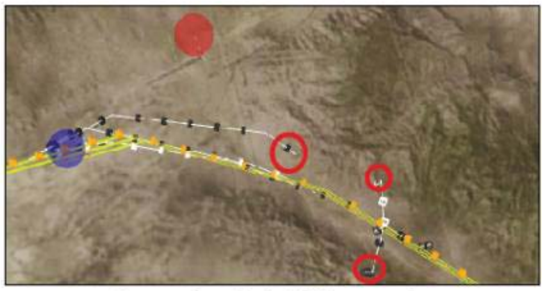

f) Top view fuel efficiency paths

Figure 45. Alternate paths generated without terrain information. Areas circled in red show where the paths violate the terrain boundary.

Figure 46 shows the same situation with terrain information added to the PSO optimization through the terrain cost component. The same tendency is seen in the paths to want to go under the aerial threat as was seen in the case with no terrain information. Now, the additional terrain cost keeps the paths from violating the terrain boundary; instead, they spread out more to the side of the threat. In this case, none of the paths violate the terrain.

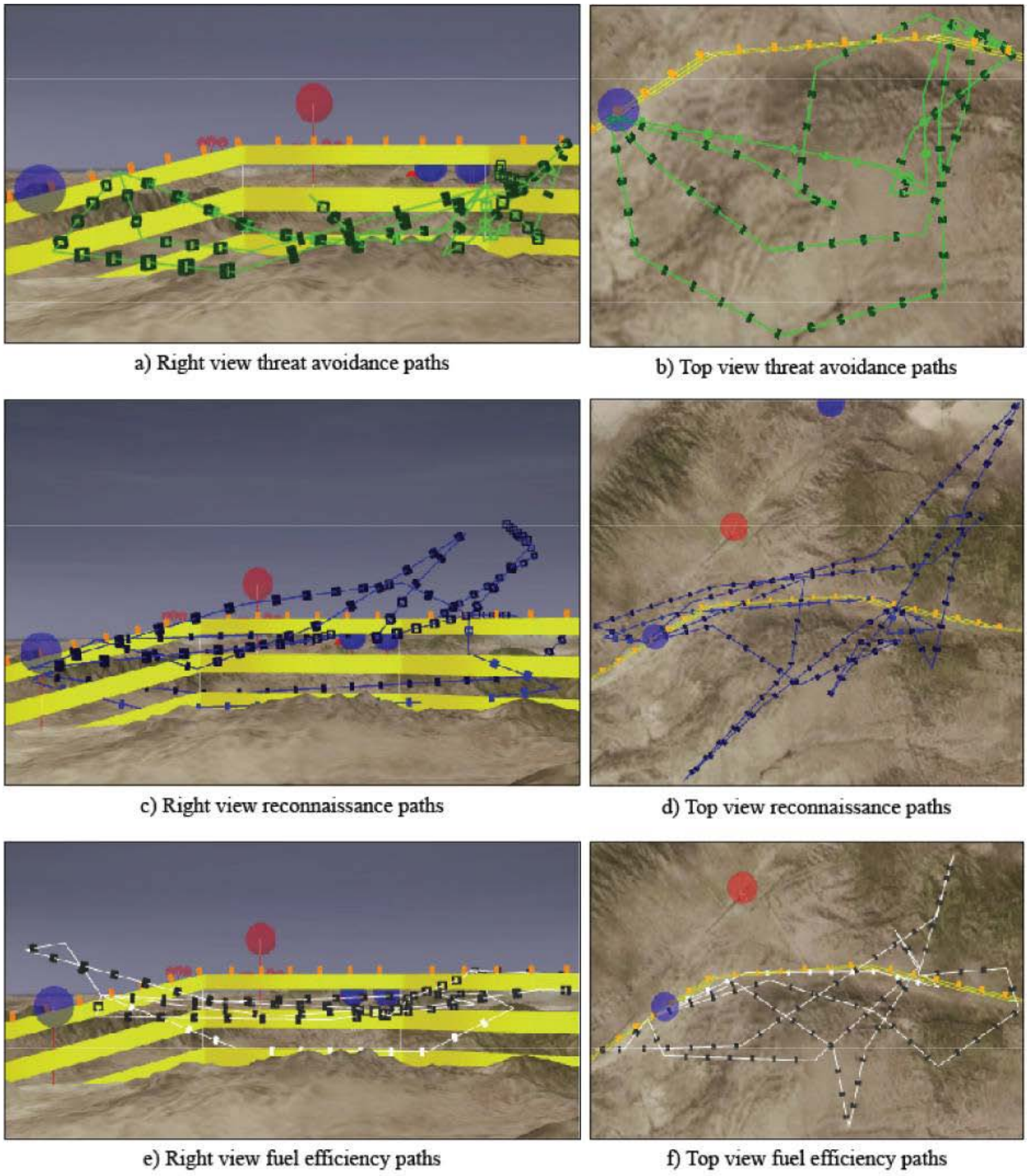

a) Right view threat avoidance paths
b) Top view threat avoidance paths
c) Right view reconnaissance paths
d) Top view reconnaissance paths
e) Right view fuel efficiency paths
f) Top view fuel efficiency paths

Figure 46. Alternate paths generated with terrain information

Figure 47 presents a an alternate path selected from each case. On the left is a path selected from those that did not use terrain information. It is clearly infeasible as it is hidden below the terrain. The path on the right was generated with terrain information and follows the same general trend but does not violate the terrain boundary and so remains feasible.

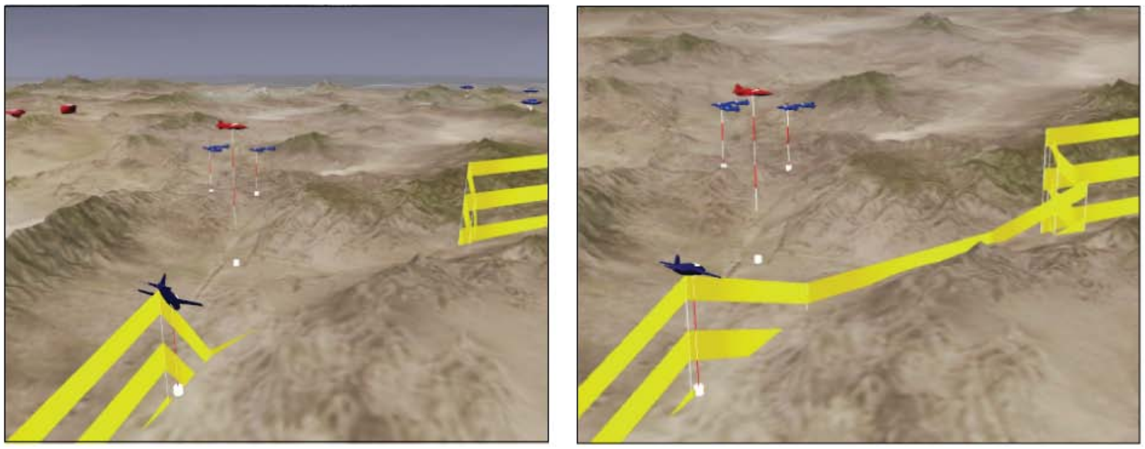


Figure 47. Updated paths; left without terrain and right with terrain.

The second scenario added an extra element of complexity. There is an aerial threat as before with an added surface-to-air missile site positioned along the UAV's path. This is a multiple-threat-zone problem. Figure 48 shows the scenario with the UAV approaching the threat location.

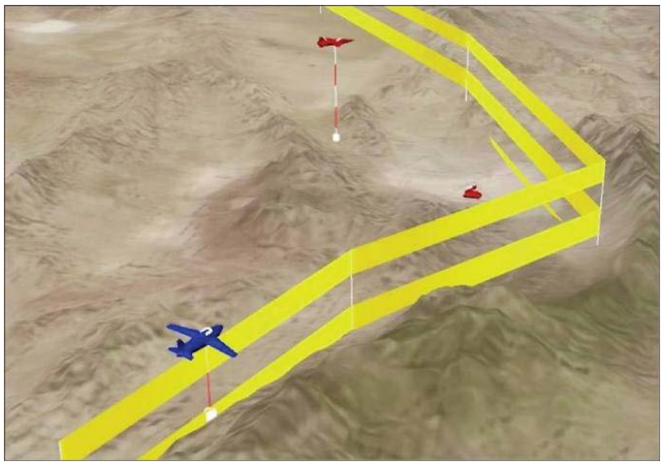
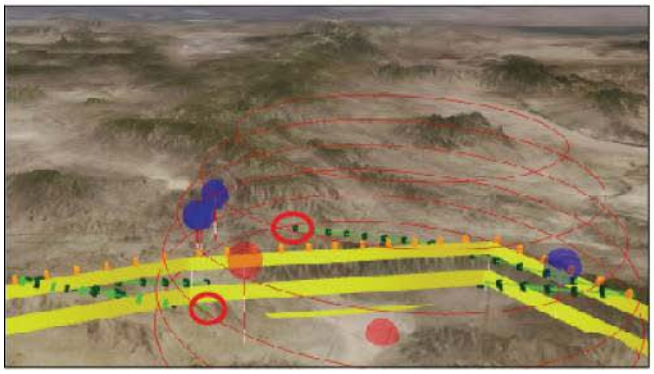
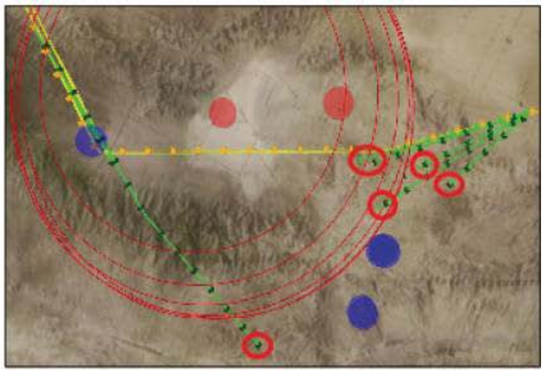
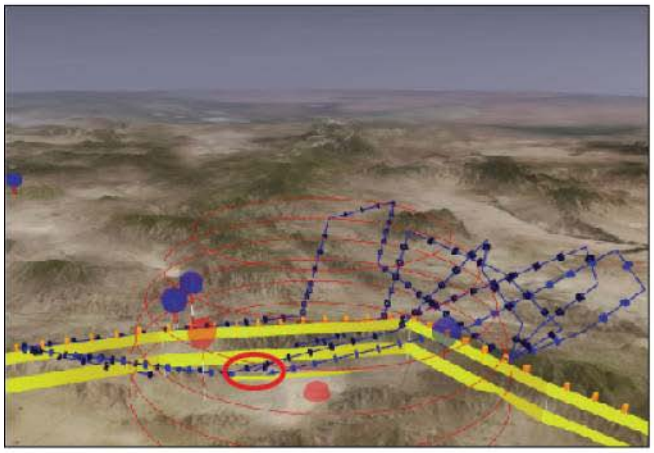


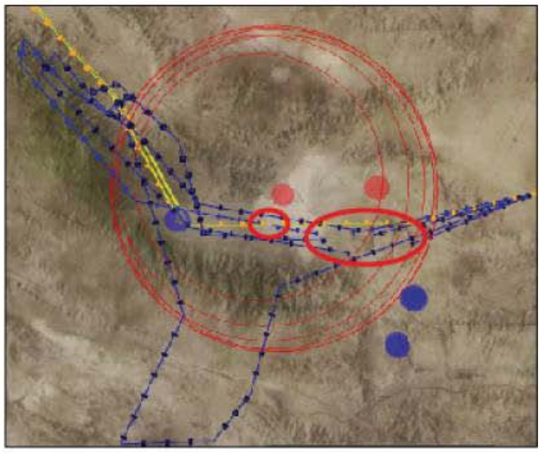
Figure 48. Original path with an aerial threat and a ground threat

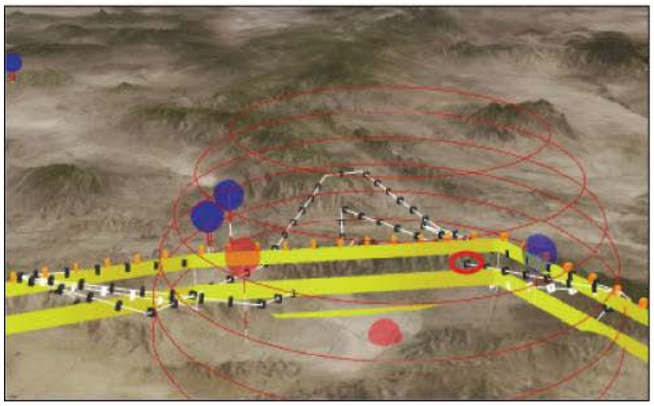
As before, paths were generated both with and without terrain information. The paths generated without terrain data are presented in Figure 49. Here one of the paths would actually be feasible. All other paths violate the terrain at some point. The terrain violations are noted by red circles.


a) Left view threat avoidance paths


b) Top view threat avoidance paths


c) Left view reconnaissance paths


d) Top view reconnaissance paths


e) Left view fuel efficiency paths

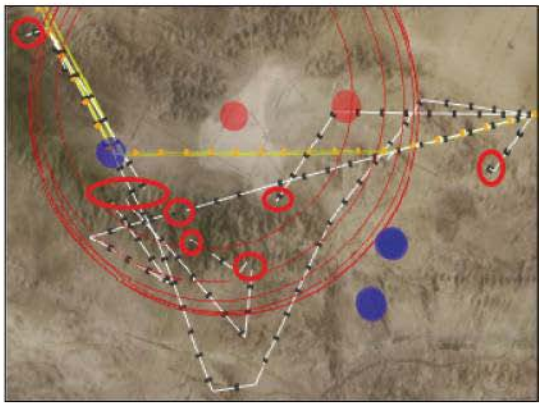

f) Top view fuel efficiency paths

Figure 49. Paths generated without using terrain information. Areas circled show sections of the paths that violate the terrain boundary.

Among the paths found with terrain information, there were no ground collision issues, as seen in Figure 50. The results in this case are very similar to those obtained with the first test scenario.

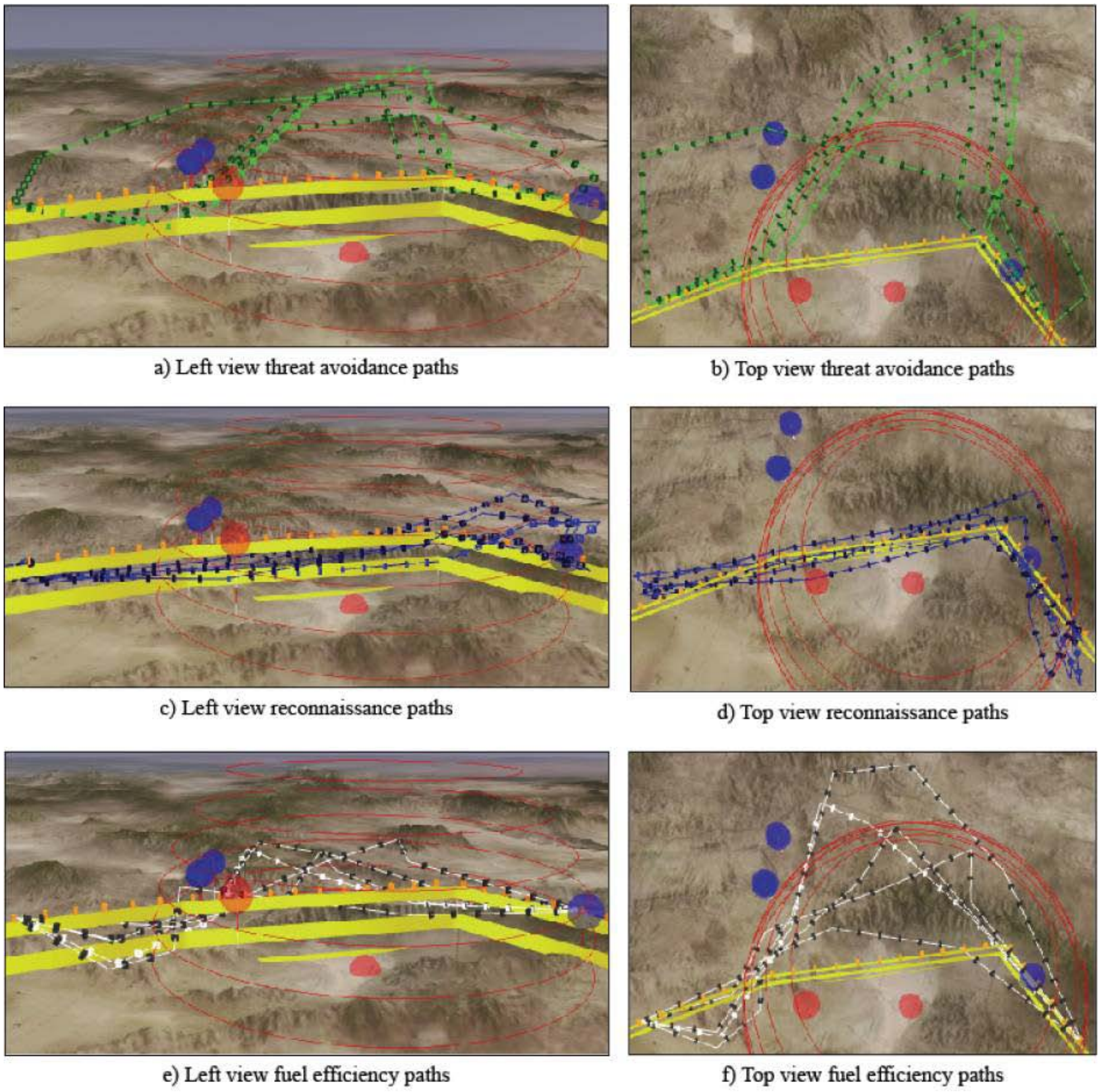

a) Left view threat avoidance paths
b) Top view threat avoidance paths
c) Left view reconnaissance paths
d) Top view reconnaissance paths
e) Left view fuel efficiency paths
f) Top view fuel efficiency paths

Figure 50. Paths generated with terrain information.

The path visualizations (presented for two test cases) demonstrate how adding terrain information to the cost function is effective in preventing infeasible paths. They also demonstrate that the terrain cost component does not fundamentally alter the type of path that is optimal. In both cases, the optimal paths tend to follow similar trajectories. The terrain cost component simply moves the cost function minimum from infeasible space below the terrain to feasible space above the terrain.

Using terrain information has a cost associated with it. It takes longer to generate the alternate paths when terrain information is included because for every iteration, path, and path point the algorithm must get the terrain height and compute that component of the cost. Figure 51 helps quantify this tradeoff. For each test scenario, the time to complete the calculations is presented, with the blue bar being the time without using terrain information and the red bar being the time with terrain information. The green bar is a count of the number of infeasible paths when calculations are performed without terrain information and the purple bar is a count of infeasible paths when terrain is included. This was repeated five times for each of the six scenarios and the values were averaged to produce the data in the graph.

Adding terrain information to the optimization increases the calculation time by two seconds on average. In some of the cases, that meant doubling the planning time. This takes away time the operator previously had for selecting a path. Yet, more path selection time is of no benefit if there are not feasible paths available to select.

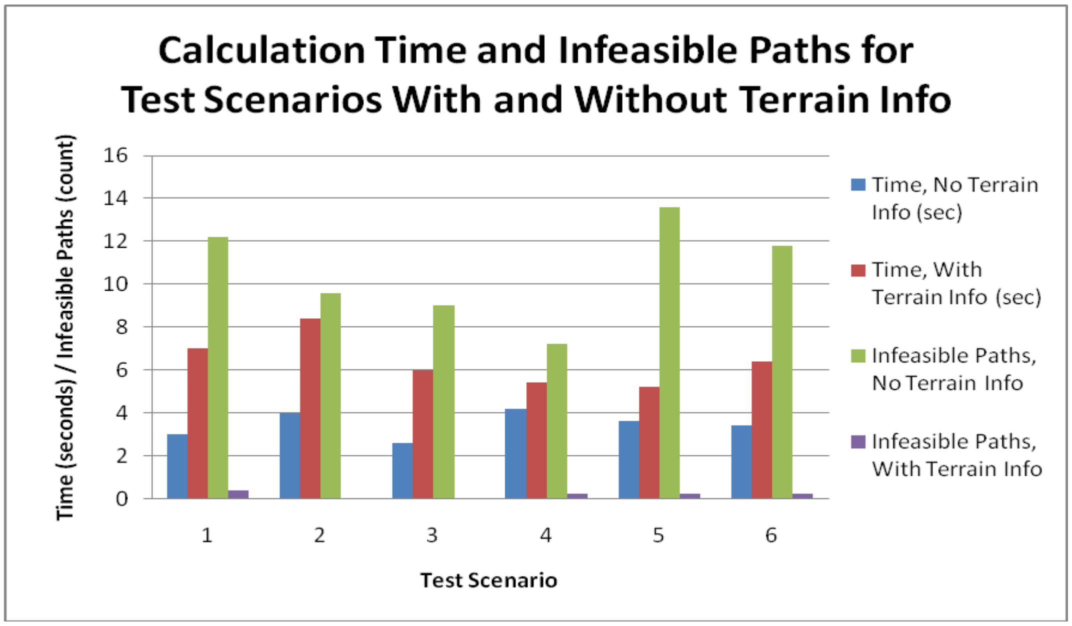


Figure 51. Calculation time and infeasible path count averages for six scenarios

On average, without terrain information, the majority of the paths were infeasible. With terrain information, occasionally one of the 15 paths will be infeasible. This happens because the algorithm attempts to return five unique paths from one PSO optimization and the paths are not forced to be feasible. If PSO does not find enough feasible paths that are unique, it may return an infeasible path to the operator. This is not a problem as it occurs rarely and the operator can simply ignore the infeasible path in favor of better solutions.

Meta-paths as a Decision-Making Tool

The meta-paths were tested on the same scenarios used for testing the cost functions earlier in this chapter. For each scenario, three sets of alternate paths were generated. The weights for the objectives for each of these situations were the same as in the earlier cases. Before the path planning code is run, the weights can be adjusted, but once the algorithm is running, the values are fixed.

The first test case, presented here, demonstrates how the meta-paths approximate the underlying paths. The meta-paths are shown in Figure 52. They are colored according to the underlying paths that compose them (white for fuel efficiency, blue for reconnaissance and green for threat avoidance).

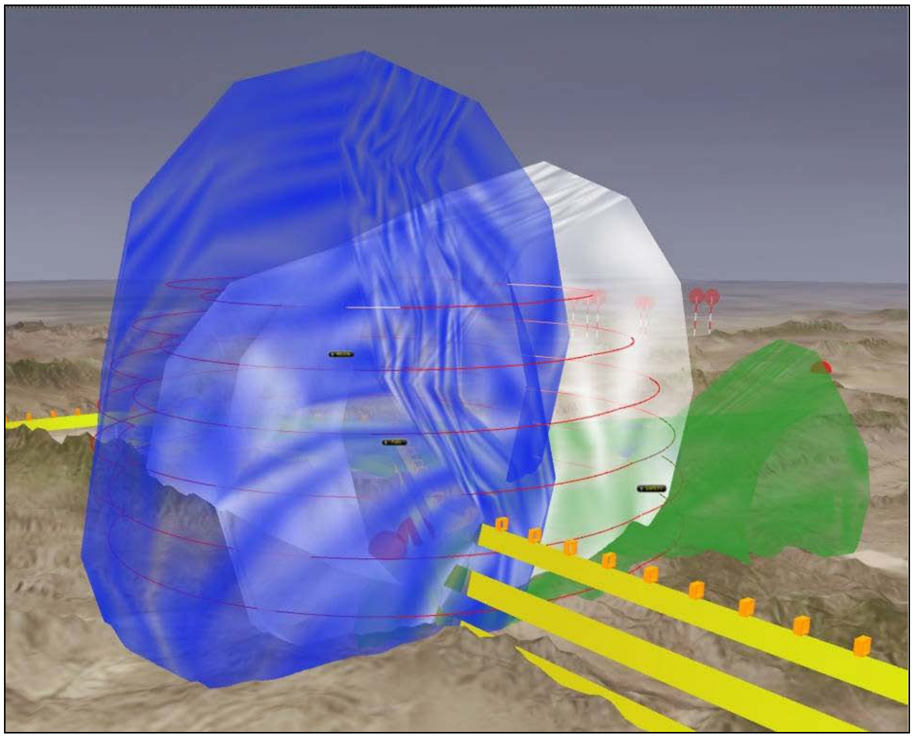


Figure 52. Meta-path representation of 15 alternate paths

The meta-paths in Figure 52 reveal some of the general characteristics of the individual paths from which they were composed. The individual paths (viewed from the same location as the meta-paths) are displayed in Figure 53. According to the meta-path representation, the threat avoidance paths all go to the right of the threat and tend to stay safely outside of the threat zone. The reconnaissance meta-path is larger and is caused a more spread out set of paths. The fuel efficiency paths tend to take the middle ground between the other sets of paths, which also results in the shortest paths.

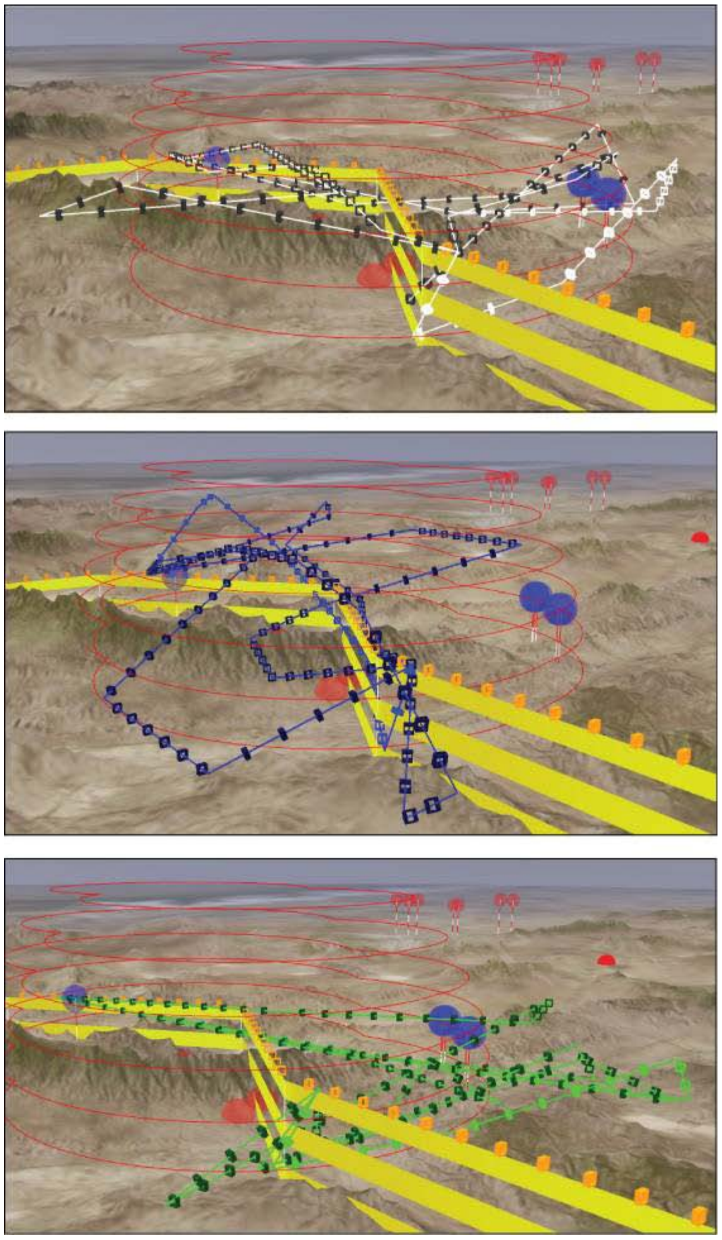


Figure 53. Individual paths for fuel efficiency (top), reconnaissance (middle), and threat avoidance (bottom)

The trends conveyed by the meta-paths are useful in allowing the operator to have a general overview of the alternatives without having to look at all the alternate paths at once. In this way, the operator can quickly understand the tradeoffs that are available and then spend more time looking at the individual paths that are of most interest. For instance, if safety is the operator's top concern, the threat avoidance paths would seem to be the best set to investigate in more detail.

A second case is also presented (see Figure 54), showing again how the meta-paths provide an intuitive intermediate layer in the path-selection process. In this case there is just one ground threat the UAV must avoid.

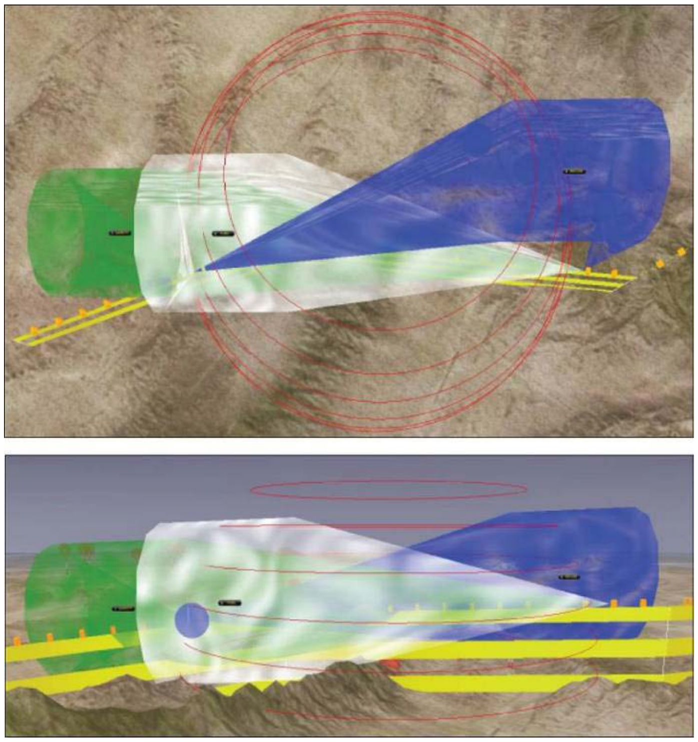


Figure 54. Meta-paths for the second test case, with a top view and a side view given

According to the meta-paths, all the paths stay relatively close to the original path. The fuel efficiency and threat avoidance paths are initially more spread out and then come together as they move toward the end point of the path segment. The reconnaissance paths exhibit the opposite behavior. Figure 55 presents a top view and a side view of each of the sets of paths that were generated. These results bear out the assumptions made from looking at the meta-paths and interpreting them as clouds containing the individual paths.

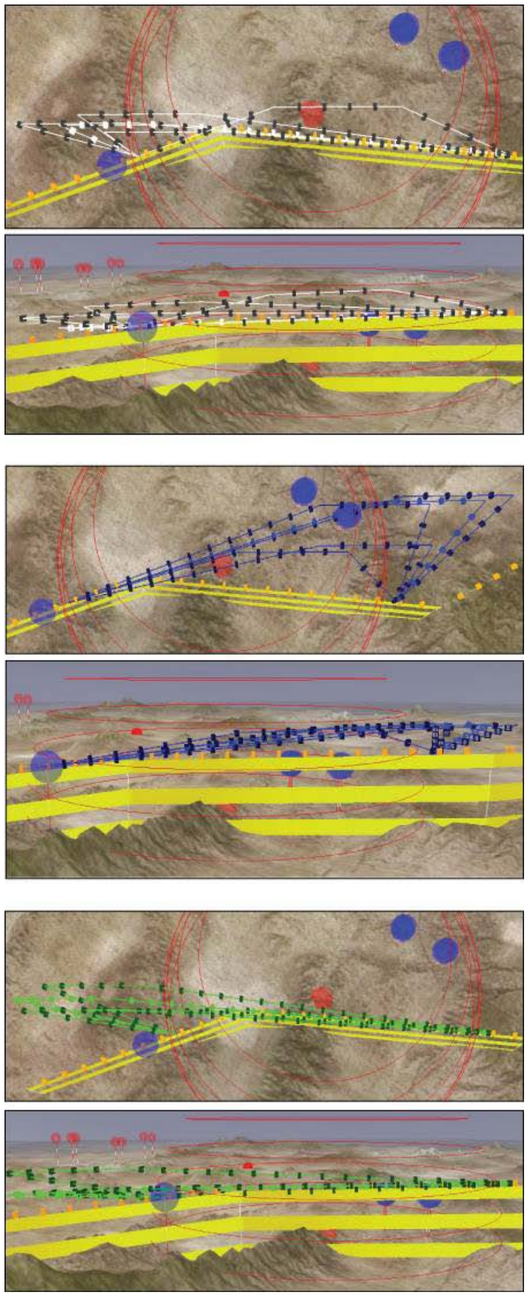


Figure 55. Side view and top view for alternate paths for fuel efficiency, reconnaissance and threat avoidance

Relative Fitness and Component Fitness Bars

Relative fitness provides a numerical means for the operator to evaluate the alternate paths, in addition to the visual representation. The optimum value for a path is 100 and less-than-optimal solutions have smaller values. Thus, the operator can weigh the optimality of the solutions against their trajectories to make a better decision for how to reroute the UAV.

Component fitness bars are a visual means of identifying tradeoffs between alternate path choices. They provide a more detailed gauge of the optimality of the paths by treating each objective cost separately. It is normal to have cases where paths have similar relative fitness values but achieve that fitness in different ways. For example, one path might be very good in reconnaissance while another is good in fuel efficiency.

Two cases will be analyzed to show how these additional measures play an important role in the decision-making process. The first case is one with an aerial threat and a ground threat obstructing the UAV's original course. The meta-paths generated for this situation are presented in Figure 56.

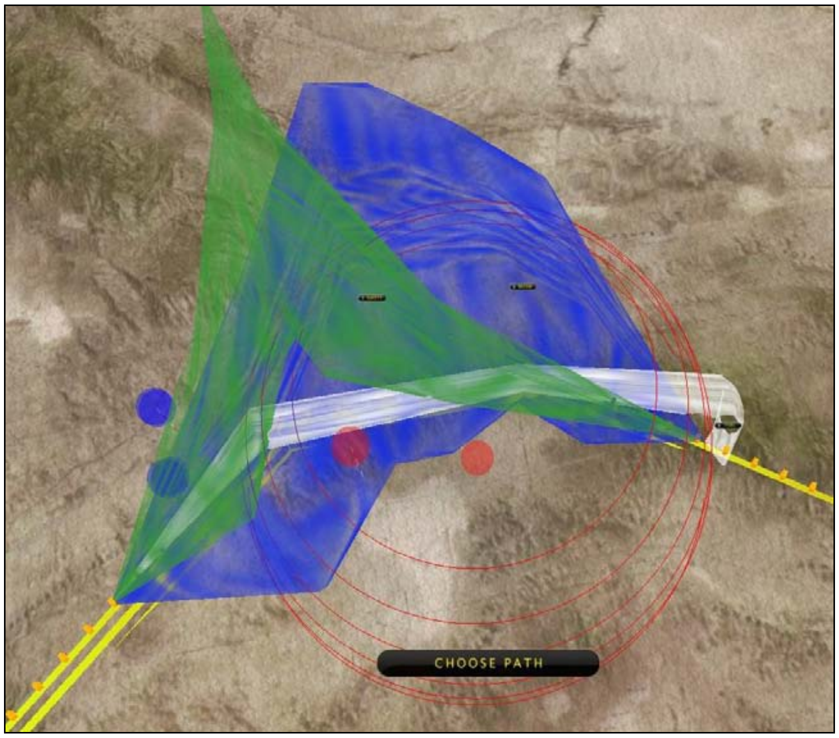


Figure 56. Meta-paths generated for test scenario

Figure 57 shows the alternate paths for threat avoidance that the operator could choose from. When one of those paths is highlighted, its relative fitness and component fitness bars are displayed. It is noted immediately that the optimal path (with a value of 100 for relative fitness) is much better in reconnaissance than the other alternate paths. The reconnaissance cost for that path would be very low so that even though the other paths have good reconnaissance values, they are relatively quite large. This is one disadvantage to computing fitness values relative to the best value – the results can be skewed by one very good path.

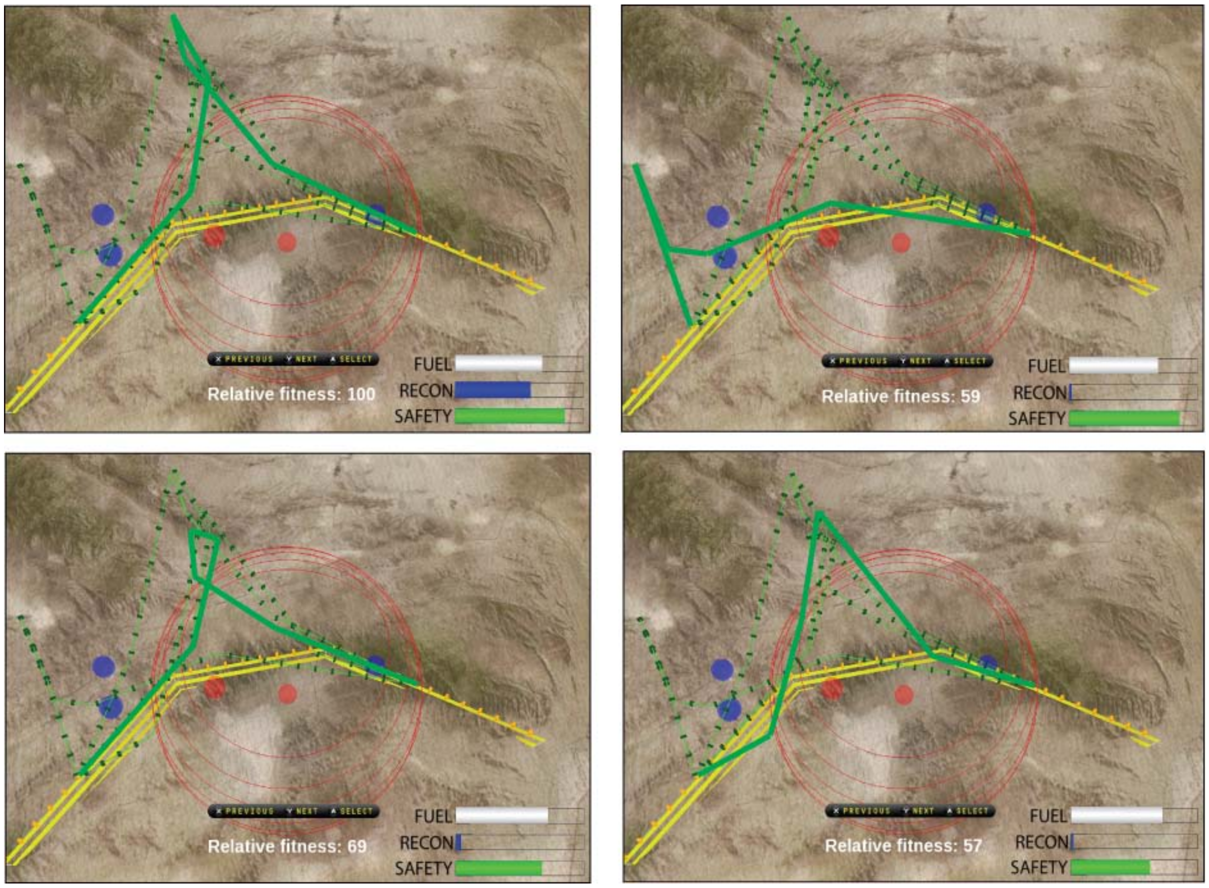


Figure 57. Four threat avoidance path choices with relative fitness values and component fitness bars

The values used to create the component fitness bars are given in Table 4. It is actually the third path that has the best threat avoidance cost of any of the alternatives. This illustrates why returning multiple paths is advantageous for finding one that will meet the operator's needs. If threat avoidance was the top priority for this mission, path 3 would be a good choice to consider.

Table 4. Component relative fitness values used to create the component fitness bars for the threat avoidance paths

Threat Avoidance Paths

Path	Fuel Relative Fitness	Recon Relative Fitness	Threat Relative Fitness
1	68.3	59.2	86.2
2	72.4	3.7	67.4
3	73.9	1.7	100.0
4	69.6	1.6	87.1
5	72.5	2.1	62.9

If the extra wiggles in that alternate path are undesirable, then the path shown in the lower right of the figure (with a relative fitness of 57) may be a good option. Like the other, more highly-rated options, it swings wide of the aerial threat while still passing over the reconnaissance points. Evaluating tradeoffs in this manner demonstrates the need to balance the visual information with the numerical fitness values in order to select the best option.

For comparison, Table 5 presents the relative fitness values for all the paths generated by PSO for this test scenario. Among the fuel efficiency paths, there are 3 that have a value of zero for relative fitness. This is a case where the algorithm did not find five good unique alternative paths. Ideally, these undesirable solutions would not be returned but that functionality has not yet been built into the algorithm.

Table 5. Relative fitness values for all 15 alternate paths

Fuel Efficiency		Reconnaissance		Threat Avoidance	
Path	Relative Fitness	Path	Relative Fitness	Path	Relative Fitness
1	100.0	1	100.0	1	100.0
2	72.9	2	64.4	2	69.2
3	0.0	3	42.8	3	65.6
4	0.0	4	42.1	4	59.6
5	0.0	5	40.1	5	57.9

The test case just presented focused on the types of solutions the path planner might generate as well as how the relative fitness and fitness component bars factor into the decision process. The second test case is an example path planning scenario where all these features are used to choose an alternate path to re-task the UAV. It is designed to explain the decision-making process that would be involved in such a situation.

Figure 58 presents a flow diagram of a path planning process incorporating meta-paths, relative fitness, and fitness component bars. Initially, an alert pops up, telling the operator that the UAV needs to be re-tasked. PSO runs and returns alternate paths which are grouped into meta-paths and presented to the operator for investigation. The UAV is low on fuel, so the fuel efficiency meta-path is selected. The paths in that category are displayed, with the optimal path highlighted. After using multiple views to look at the paths, the operator decides that staying a bit farther from the threat and selects the path with a relative fitness of 73. In doing so, the operator is sacrificing reconnaissance, which is the least important objective for this mission. The UAV is then re-tasked, ending the path planning process.

This sample scenario demonstrates the types of decisions the operator would have to make to in a path planning situation and the flexibility of having multiple alternate path choices.

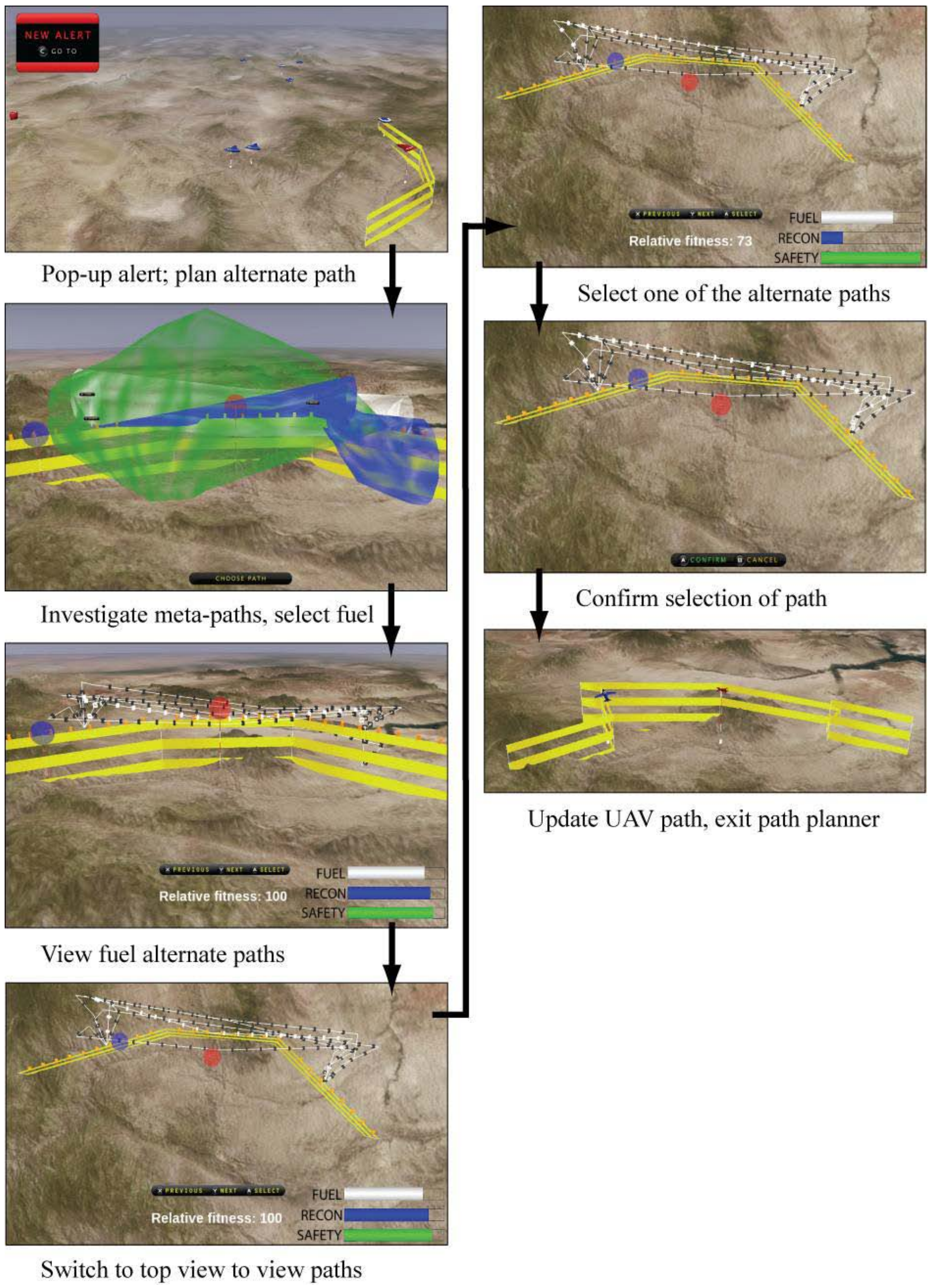

NEW ALERT
GO TO
Pop-up alert; plan alternate path
CHOOSE PATH
Investigate meta-paths, select fuel
PREVIOUS NEXT SELECT
Relative fitness: 100
FUEL
RECON
SAFETY
View fuel alternate paths
PREVIOUS NEXT SELECT
Relative fitness: 100
FUEL
RECON
SAFETY
Switch to top view to view paths
PREVIOUS NEXT SELECT
Relative fitness: 73
FUEL
RECON
SAFETY
Select one of the alternate paths
CONFIRM CANCEL
Confirm selection of path
Update UAV path, exit path planner

Figure 58. Flow diagram for the decision-making process involving meta-paths, relative fitness, and component fitness bars

For reference, Table 6 presents the relative fitness values for all the paths generated in this test case. The reconnaissance paths are the most similar in terms of relative fitness. This would be expected as these paths tend to be more closely clustered to pass over the reconnaissance targets. The threat avoidance values are characteristically the most spread out as there are many ways to maneuver around the threat, some which have significantly better costs than others.

Table 6. Relative fitness values for all the alternate paths

Fuel Efficiency		Reconnaissance		Threat Avoidance	
Path	Relative Fitness	Path	Relative Fitness	Path	Relative Fitness
1	100.0	1	100.0	1	100.0
2	85.5	2	96.9	2	53.1
3	73.1	3	92.4	3	47.3
4	70.3	4	88.0	4	40.4
5	69.7	5	84.7	5	40.3

Interactive Optimization and Weight Adjustment as a Decision-Making Tool

The interactive weight adjustment method of interaction is demonstrated using a simulated scenario. Figure 59 presents an example case of how path planning might be performed with this algorithm. Initially, an alert pops up and the operator investigates it. The operator has the ability to adjust the weights and then run PSO to generate an optimal path using those weights. If the characteristics of the alternate path are not desirable, the operator can re-plan as many times as necessary, altering the weights each time. In this simulation, it takes four re-planning phases to generate an alternate path that satisfies the operator. That path is used to re-task the UAV.

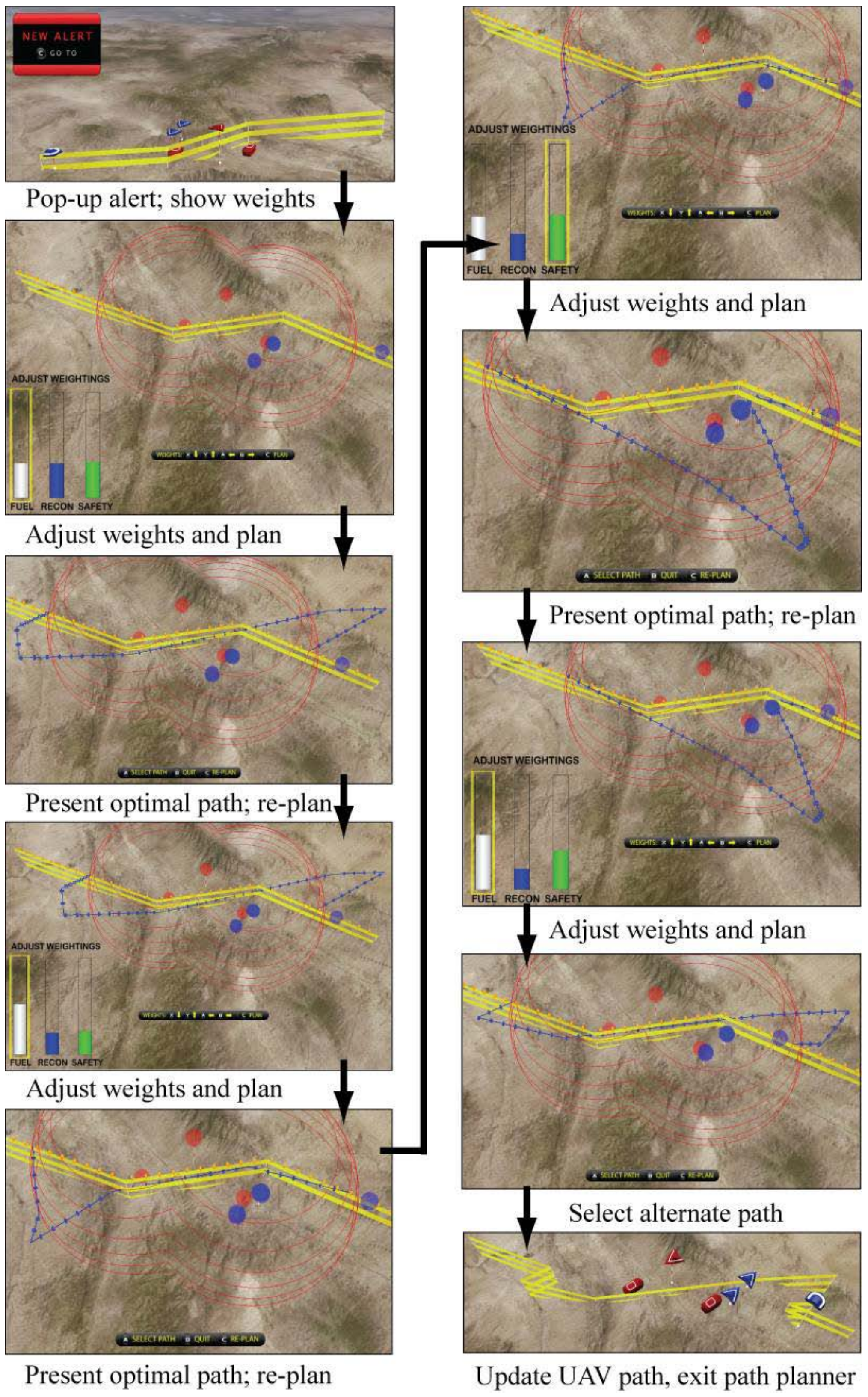


Figure 59. Path planning flow diagram using interactive weight adjustment

Table 7 presents the timetable of actions for this planning process. The operator spends nine seconds starting the path planning process and maneuvering to a good location to view the paths. On average, each iteration (involving a weight adjustment, PSO optimization, and path viewing) took nine seconds to complete. The operator finished planning after 38 seconds.

Table 7. Timetable for path planning process with interactive weight adjustment

Action	Time of Occurrence (sec)
Start planning, 1st run	9.4
PSO completes optimization, 1st run	9.7
Decide to re-plan path	11.8
Adjust fuel efficiency weight	13.9
Start planning, 2nd run	18.1
PSO completes optimization, 2nd run	18.3
Decide to re-plan path	20.0
Adjust reconnaissance weight	22.8
Adjust threat avoidance weight	24.6
Start planning, 3rd run	25.8
PSO completes optimization, 3rd run	26.6
Decide to re-plan path	30.3
Adjust fuel efficiency weight	34.3
Start planning, 4th run	36.1
PSO completes optimization, 4th run	36.5
Decide to keep path	38.3
Done with path planner	38.3

For this same scenario, a second run was conducted to see if the timings seen in the first test were consistent. The second test proceeds in a very similar fashion to the first, as seen in Table 8. The operator conducts one extra optimization and takes 46 seconds to re-task the UAV. In all cases, the PSO algorithm returns the optimal solution in 0.5 seconds.

Table 8. Test run 2 for interactive weight adjustment case

Action	Time of Occurrence (sec)
Start planning, 1st run	8.6
PSO completes optimization, 1st run	8.9
Decide to re-plan path	13.8
Adjust fuel efficiency weight	14.9
Start planning, 2nd run	16.4
PSO completes optimization, 2nd run	17.1
Decide to re-plan path	20.0
Adjust fuel efficiency weight	22.2
Adjust reconnaissance weight	23.8
Adjust threat avoidance weight	25.6
Start planning, 3rd run	28.1
PSO completes optimization, 3rd run	28.8
Decide to re-plan path	31.6
Start planning, 4th run	33.7
PSO completes optimization, 4th run	34.4
Decide to re-plan path	36.1
Adjust fuel efficiency weight	39.9
Adjust reconnaissance weight	40.9
Start planning, 5th run	42.8
PSO completes optimization, 5th run	43.3
Decide to keep path	45.7
Done with path planner	45.7

It took the operator (who was familiar with the system) 38.3 and 45.7 seconds respectively to complete the two trial runs. This is more than the built-in time limit. That means that further enhancements would be needed before this interaction technique would be a viable alternative.

Post-Planning Objective Weight Adjustment as a Decision-Making Tool

Post-planning objective weight adjustment is one of the modifications to the path planning algorithm to allow the operator to control the weights for the different objectives to investigate paths that better fit with the operator's goals. Following the PSO optimization (still based on pre-defined weights for fuel efficiency, reconnaissance, and threat avoidance), the interface shown in Figure 60 is presented to the operator.

The bars in the lower left are adjustable by the operator. Initially, the weights for each objective are set to be equal. The black menu reminds the operator of how the weights are adjusted. The highlighted bar is the one that is currently being changed. Adjusting the bars has a one-to-one correspondence to changing the component weight coefficients. Increasing one value will automatically decrease the others and vice versa. This is for interface transparency; so the operator knows precisely what values are being used.

Changing the weights automatically updates the paths that are displayed. As can be seen in the figure, there is an intuitive correspondence between the paths and the weights. When fuel efficiency is increased, the paths tend to move toward the original path. Increasing reconnaissance ensures that the paths pass over the waypoint locations. When threat avoidance is more heavily weighted, the paths tend to go around the threat dome instead of cutting through it.

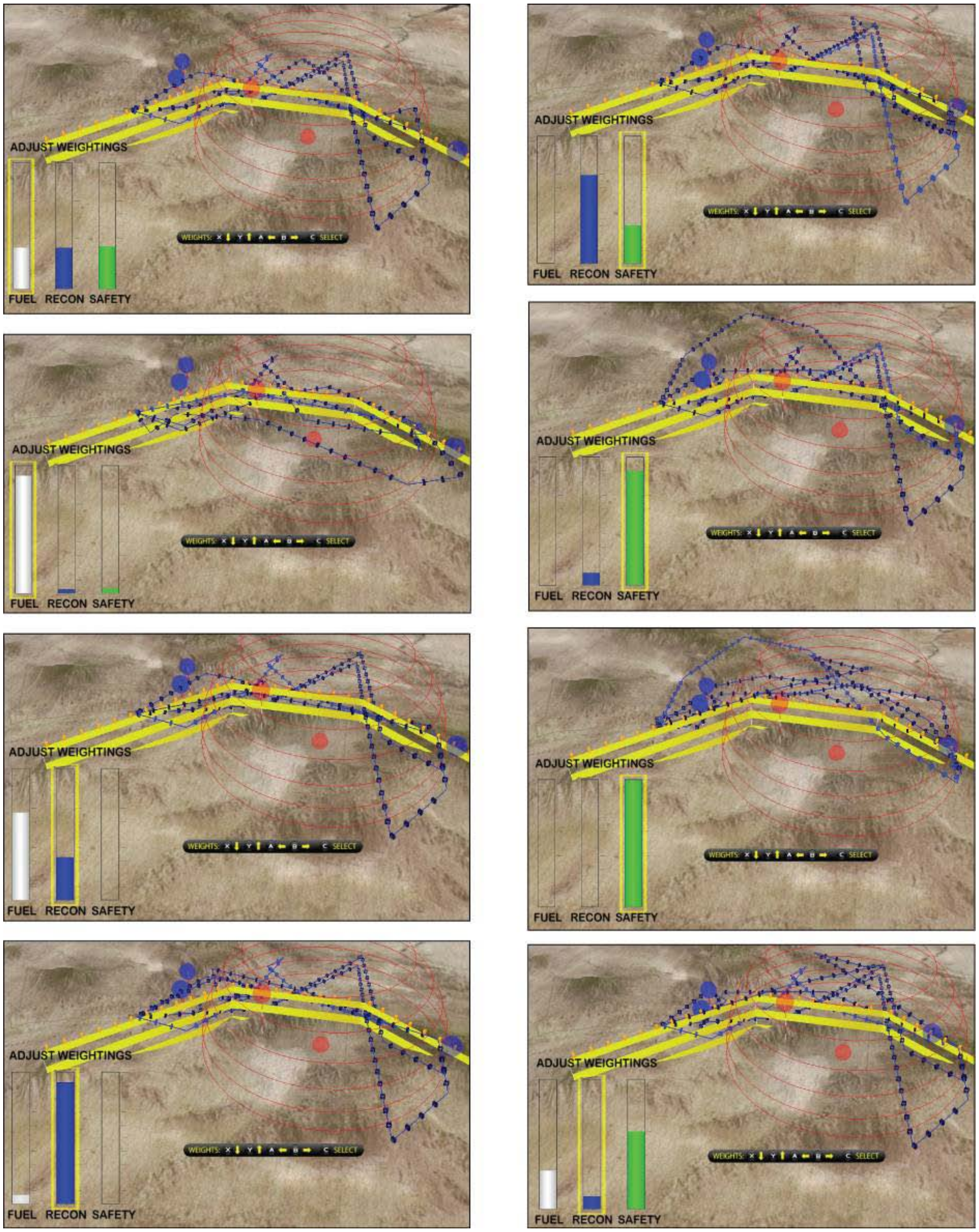


Figure 60. Selected paths returned by different weightings for operator involvement

In Figure 61, there is a flow diagram to demonstrate how this interaction method would be used in UAV path planning. Once the operator initiates the path planning process, PSO finds and returns alternative paths for analysis. The top paths are shown for an equal weighting of fuel efficiency, reconnaissance, and threat avoidance. The operator then adjusts the weights to fit personal preferences and mission objectives. In this case, safety was considered most important, followed by reconnaissance and then fuel efficiency. The screenshots show how the paths change when the weights are adjusted by the operator. Instead of following the original path as closely, the paths now move to the right of the original path to avoid the threat. When the operator is satisfied with the weights and the paths that are displayed, the selection is locked. Then the operator can investigate the alternate paths in more detail with the help of the component fitness bars. The picture in the top right shows one of the paths that does well in all the objective areas, striking a good balance between threat avoidance and staying close to the targets. The operator then selects the path and confirms it to update the course of the UAV.

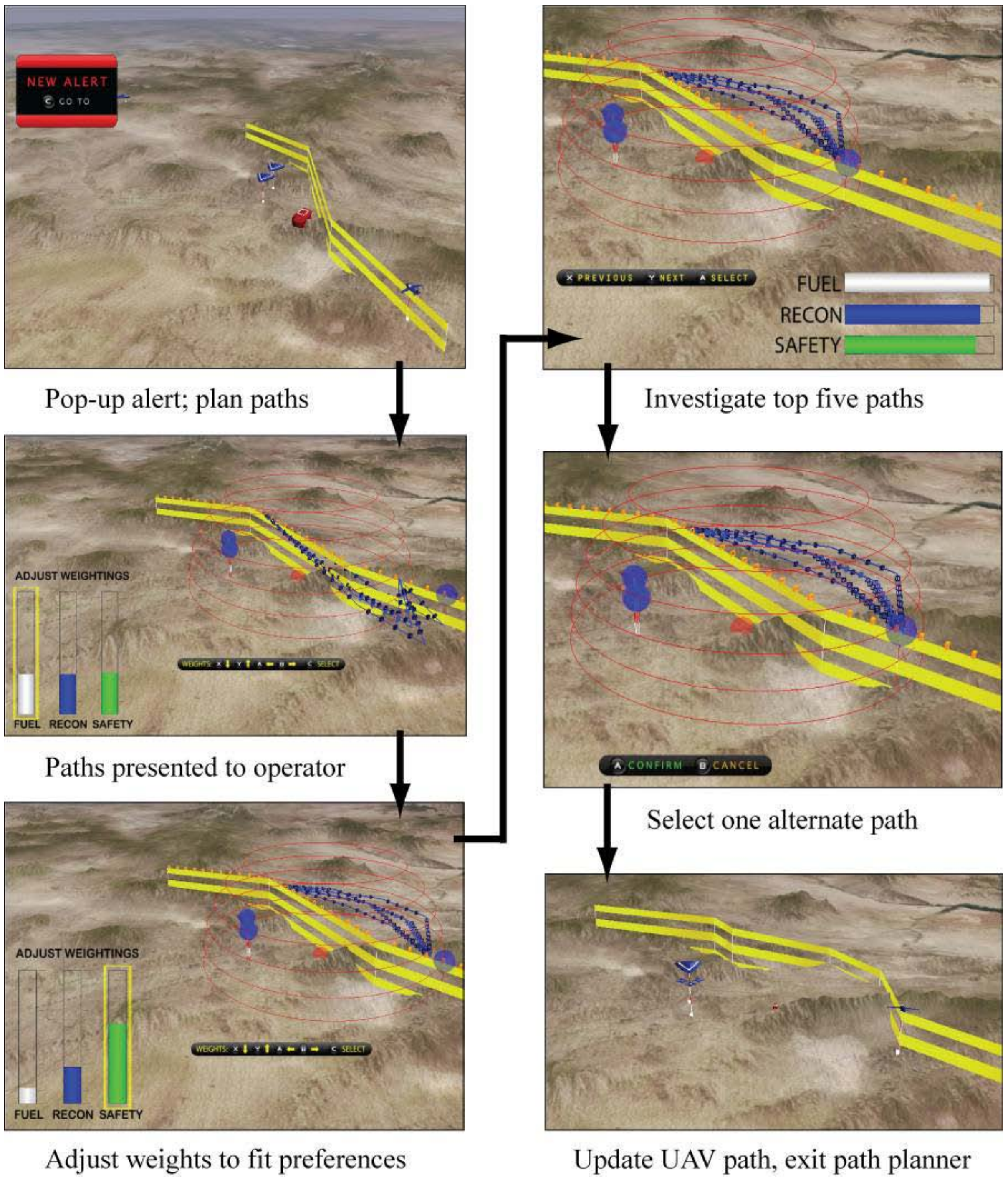


Figure 61. Post-analysis weight adjustment used in selecting a path to re-task the UAV

The operator could spend minutes adjusting weights and investigating alternate paths, but only a short period of time is allotted for the path planning process. Thus, the optimization, weight adjustment, and path selection all need to happen quickly and efficiently. Table 9 shows the timings for the test scenario given in Figure 61. The path planning algorithm took 1.2 seconds to execute and returned 76 unique paths for further analysis. Ten seconds were spent navigating in 3D space to find a good viewing angle for looking at the paths. Then, twelve seconds were spent adjusting the weights to find a good combination. The operator then switched to path investigation mode about 25 seconds into the planning process. It took four seconds for the operator to decide which of the five alternate paths to use. The UAV was then re-tasked appropriately, with the whole process taking 31 seconds.

Table 9. Timing for path planning process with adjustable weights

Action	**Time of Occurrence (sec)**
Start planning paths	0.0
PSO completes optimization	1.2
Adjust fuel efficiency weight	13.2
Adjust reconnaissance weight	16.6
Adjust threat avoidance weight	22.0
Switch to path investigation mode	24.9
Select path	29.5
Done with path planner	30.9

The planning exercise presented above demonstrates that it is possible to use post-planning weight adjustment to allow the operator to incorporate personal preferences and mission objectives into the planning process. Yet, the whole allotted time was needed to re-task the UAV. Improvements to the interaction or operator training could help speed up the process.

Chapter 5 – Evaluating Interactive Path Planning as a Decision Making Tool

Modified from a paper submitted to *Interservice/Industry Training, Simulation, and Education Conference (I/ITSEC) 2009*

Levi Swartzentruber, William Marsh, Jung-Leng Foo, James Oliver, Stephen Gilbert, and Eliot Winer[1]

Abstract

Effective command and control of unmanned aerial vehicles (UAVs) is an issue under investigation as the military pushes toward more automation and incorporation of technology into their operational strategy. Even with the advanced capabilities of UAVs, human intervention is still critical to the vehicle's response in unexpected situations. This paper presents a user study to gauge how different visual capabilities affect operator performance when using an interactive path planning tool. The path planner uses Particle Swarm Optimization (PSO) to generate alternate paths considering three objective criteria: 1) minimizing risk due to enemy threats, 2) minimizing fuel consumption, and 3) accounting for reconnaissance targets. Paths are three-dimensional B-splines, optimized based on

[1] Levi Swartzentruber and William E. Marsh were the principal authors of this study. As part of his master's work Levi developed the path planning algorithm and visualizations and aided in the running of the study. William generated the experimental design and aided in the analysis of the results. The other authors were instrumental in shaping the experiment and providing feedback and guidance throughout the process.

weightings for the three objectives while also avoiding terrain collisions. Experiment one of the user study was designed to assess three distinct display techniques according to user performance on two tasks where an alternate path must be determined. Two of the display techniques were 2D, with height data given by numerical values or glyphs, respectively, at the waypoints. The third technique provided the user with multiple 3D views to evaluate the paths. The goal of the user was to pick the path closest to the original path. In experiment two, the same 2D numerical and 3D display techniques were used. The user was to select the best path, based on the mission briefing, from a number of alternatives generated by the path planning algorithm. Each user's results were compared to a path generated by an expert who was given the same briefing. In each case, instructions were given about how to choose the path; such as locating the best threat avoidance path while still acquiring reconnaissance information. The users were evaluated on the path chosen and the time to do so. The analysis of results and concepts for further research will also be discussed.

Introduction

Military operations increasingly incorporate unmanned systems such as unmanned aerial vehicles (UAVs) into operations to perform dangerous missions and reduce human casualties. In Operation Enduring Freedom and Operation Iraqi Freedom, almost 400,000 flight hours have been logged by UAVs, not including hand-launched systems (Barry & Zimet, 2001). According to the Department of Defense's Unmanned Systems Roadmap 2007-2032, these vehicles will be expected to perform a full range of mission tasks by 2030 (US Department of Defense, 2007). At the same time, one operator will be expected to control multiple vehicles. UAVs will need to become increasingly autonomous to meet these performance requirements. However, that will not eliminate the need for human involvement, but will necessitate more efficient interaction between the human and the vehicle. Operators will move into supervisory roles, monitoring and guiding the activities of UAVs through computer interfaces rather than mechanically controlling their movements.

In a supervisory role, operators will need to orient themselves quickly to the UAV's situation and then respond with an appropriate course of action. One important area of supervisory control is re-tasking a UAV when its current path is threatened. UAV path planning is a complicated task involving visualization, optimization, path planning, and decision making. Much research has been done on algorithms for generating optimal alternate paths but almost no work has been published on how to involve the operator in the planning process. The study discussed in this paper seeks to understand how best to present information to a UAV operator and how effective a novice operator can be in selecting an optimal alternate path with the help of a path planning algorithm.

Literature Review

A UAV's path can be described as a curve in 3-dimensional (3D) space. If formulated correctly, a path is a multi-modal and multi-objective optimization problem. When the UAV needs to avoid a threat, any feasible alternate path is a possible way to complete that task, and there are cases where there are multiple unique paths that would be equally preferable. The operator uses criteria (objectives), past experience, and intuition to choose the best alternative.

Objectives included in the path generation process can be strictly path-related or can incorporate the dynamics of the UAV. One path planner implements a path-related objective by selecting a trajectory to minimize radar cross section to avoid enemy missiles (Kabamba, Meerkov, and Zeitz, 2006). Another path planner incorporates four objectives: shortest path, terrain avoidance, minimum and maximum elevation, and minimum radius of curvature for the path (Hasircioglu, Topcuoglu, & Ermis, 2008). The last two objectives involve UAV dynamics. Some algorithms are more complete than others but one that accounts for all details of the mission, UAV dynamics, and environment does not exist.

Optimization

Optimization is the formal process of finding the best solution to a problem. To perform optimization, the objectives must be formulated mathematically so that any path can be evaluated according to that formula, or cost function. The algorithm then seeks to find a path that minimizes the value of the cost function--the optimal path. Simulated Annealing (Kirkpatrick, Gelatt, & Vecchi, 1983), the Genetic Algorithm (Mitchell, 1998), and Particle Swarm Optimization (Eberhart & Kennedy, 1995) are evolutionary algorithms that would be well suited for path planning. Particle Swarm Optimization (PSO) was selected from the alternatives for its ease of implementation, fewer user-defined parameters, and good performance (Hassan, Cohanim, de Weck, & Venter, 2005).

PSO was modeled on the social behavior of birds. A population of randomly initialized particles (also called design points or swarm members) explores the design space (all possible solutions) and reaches the globally optimal solution after a series of iterations. In path planning, a swarm member is one potential alternate path, modeled as a B-spline curve. Each path is evaluated based on a fitness function which quantifies how well the path meets the objectives included in the problem formulation.

Even with algorithms capable of planning the UAV's path, it is still necessary to maintain a human presence in the path planning process because the algorithm cannot take into account all aspects of the situation. An operator must be able to make a quick, informed, accurate decision in re-tasking the UAV to avoid danger and account for deficiencies in the mathematical model. Military officers generally make decisions by creating a vision for the mission outcome, generating a plan which is evaluated and refined, and issuing orders to carry out that plan. These steps, combined with intuition and experience, often lead to the best outcome (Thunholm, 2005). The path planner used in this research generates multiple alternate paths, allowing the operator to focus on evaluating and implementing the best solution.

Information Display Techniques

Information display techniques can significantly influence the operator's ability to make decisions well. The operator will be biased largely toward interactive visual information (which is absorbed most quickly) and less toward textual information (which is processed more slowly) in decision making processes (Lurie & Mason, 2007). Proper path display will leverage this bias by giving key information, such as waypoint positions, the most visual weight.

Path planning and re-routing has traditionally been done using top-down 2-dimensional (2D) interfaces however there has been a shift in research towards using more immersive, 3D displays. The term “3-dimensional” is typically used in related literature (and will be used here) to refer to a perspective scene displayed on a flat CRT screen. On these displays, all dimensions are projected onto a 2D surface, providing a 3D perspective.

These newer 3D displays have both advantages and disadvantages. All three dimensions are integrated into a single view, providing a greater understanding of the general shape of terrain. Three-dimensional displays are also capable of providing natural depth cues such as perspective, shading, occlusion, and scaling. The benefits provided by these cues have been shown to be additive and independent (St. John, Smallman, Bank, & Cowen, 2001). However, along with the benefits of integrating all three-dimensions comes a cost. All dimensions are ambiguous in a 3D perspective display. In particular, a line-of-sight (or projective) ambiguity exists, requiring the use of a drop shadow or drop line to clearly attach an object to a location on the ground plane. In addition, angles and distances are distorted in 3D displays. These distortions make 3D displays a poor choice for relative position tasks, where precision is desired. (Smallman, St. John, Oonk, & Cowen, 2001; St. John, Cowen, Smallman, & Oonk, 2001)

Two-dimensional displays have been shown to be superior for tasks requiring judgments of precise relative positions, as they accurately depict precise angles and distances. St. John et al. (2001) found that users of 2D plan views outperform those using 3D views to judge precise lines of sight across terrain. Two-dimensional displays represent each dimension faithfully, making it easy to judge distances along each dimension. The primary problem with 2D displays is that line-of-sight ambiguity does still exist on the z-dimension. This along with the difficulty of using depth cues other than occlusion, means that altitude must be displayed in another way, often as a digital readout (St. John, Cowen, Smallman, & Oonk, 2001; Smallman, St. John, Oonk, & Cowen, 2001; St. John, Smallman, Bank, & Cowen, 2001).

Current Path Planners

In many path planners, little attention is given to information visualization. The algorithm usually automatically implements the best path solution, completely removing the operator from the process. Some utilize simple 2D interfaces, completely ignoring height information, as seen in Figure 62 (Shangming, Zefran, & DeCarlo, 2008). Figure 63 shows a typical 3D visualization (Soto, Nava, & Alvarado, 2007). Again, there is no interaction and no alternative paths from which to choose.

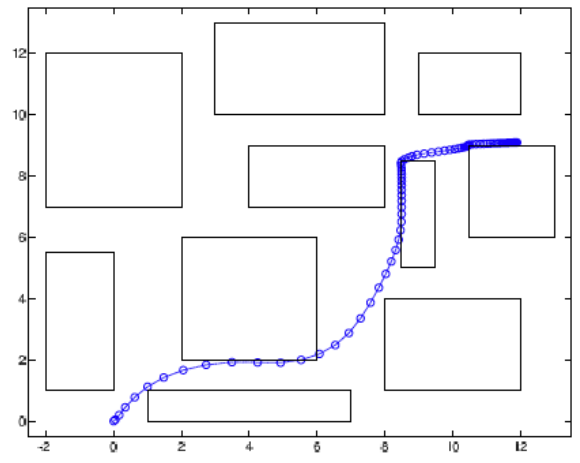


Figure 62. 2D representation of UAV path

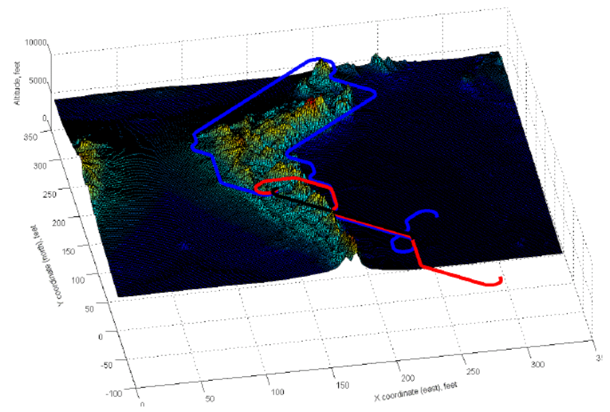


Figure 63. 3D isometric representation of multiple paths

The path planner presented here was built as a component of the Virtual Battlespace environment, show in the C6 in Figure 64. Development of the Virtual Battlespace commenced in 2000 when a research team at Iowa State University's Virtual Reality Applications Center (VRAC) began collaborating with the Air Force Research Lab's Human Effectiveness Directorate and the Iowa National Guard's 133rd Air Control Squadron. The goal of this project was to create an immersive virtual reality (VR) system for distributed mission training. Having an immersive 3D display allows the designers full freedom in how to display path information to the operators. This freedom also allows the paths to be generated natively in 3D so as to not limit the solutions the path planner can produce.

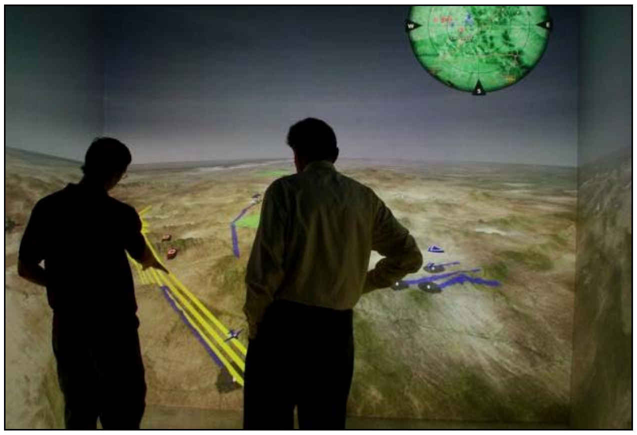


Figure 64. Virtual Battlespace in the C6

PSO Path Planner

Alternate paths presented to users in this study were generated using PSO, which is one of several heuristic methods that have been successfully implemented as a means to solve path-planning problems. To begin the process, the starting point, ending point, and any waypoints in between are identified. They form the initial design point for the problem from which a search space is defined. Next, any enemy entities, friendly entities, or obstacles within the search space are identified and a threat zone of radius R_T is generated for each of them. The affected waypoints are identified as reconnaissance targets with radius R_R. By default, the value for R_T is set to be 20,000 feet and R_R is set to be 2,000 feet, but they can be changed to suit the controller's preferences.

Formulation of the optimization cost function begins with the description of a B-spline curve to represent the path of the UAV. Figure 65 shows the original curve $p_0(u)$ in blue (that requires re-planning because it violates threat zone Z_T) and an alternate path $p(u)$ in red (that avoids the threat zone while still attempting to be within the reconnaissance zones Z_R).

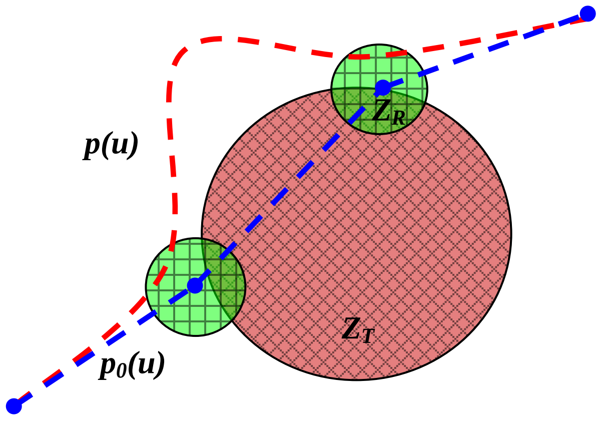


Figure 65. 2D illustration of a simple threat avoidance problem

Each path is a sequence of line segments approximating the curve. It is these segments that are used for calculating the fitness value for the path, based on the cost function. This function needs to accommodate preferences for fuel efficiency, reconnaissance, and threat avoidance. The total cost function is represented by the following equation:

$$C = K_1 C_T + K_2 C_L + K_3 C_R + T \tag{35}$$

Where C_T is the cost due to violation of the threat zones, C_L reflects the cost incurred from increasing distance of the path (i.e. using more fuel) and deviation from the original path, and C_R is the cost incurred by deviating from reconnaissance locations. The last term, T, is a cost component for flying too close to the terrain. The constants K_1, K_2, and K_3 in Eq. (35) are component weights that determine the relative emphasis placed on each component with respect to the overall cost function. The weights are normalized between zero and one and all three must sum to one. T has no weighting factor because it must be present in any arrangement to ensure path feasibility.

The threat component C_T is a scaled sum of the threat zone violations for each curve segment $p(u_i)$. The distance from the that segment to the threat is computed as $d(p, Z_T)$. The violation is scaled by the threat zone radius R_T and the number of threats S_T. The 10 is a scaling factor (determined empirically, but user defined) to keep all component costs in the same order of magnitude (and is present in the other component cost equations). The threat cost is then defined as:

$$d(p(u_i), Z_T) = R_T - [Separation\, p(u_i), threat] \tag{36}$$

$$C_T = 10 + \sum_{1}^{N} C_i * \left(\frac{1}{R_T S_T}\right) \;\; where{:}\; C_i = \begin{cases} \dfrac{d(p(u_i), Z_T)}{0}, \dfrac{d(p(u_i), Z_T) \geq 0}{d(p(u_i), Z_T) < 0} \end{cases} \tag{37}$$

The curve length component is computed using an approximation of the total curve length L. The length is the sum of all the path segment lengths. The cost is then found by dividing this value by the shortest possible path length L_0 and scaling it to be of the right order of magnitude:

$$C_L = 10 * \left(\frac{L}{L_0}\right), \;\; L = \sum_{i=0}^{N-2} |p(u_{i+1}) - p(u_i)| \tag{38}$$

The final component is for reconnaissance, C_R. The quality of the data gathered by the UAV depends on the field of view the target occupies in the UAV's sensors. This is determined by the angle θ between the vehicle and the target as well as the by the separation distance $d(p, L_R)$. For each target, only the curve segment $p(u_i)$ that comes closest to it is considered. The cost is scaled by the number of targets, N_T present. The reconnaissance cost is defined as:

$$d(p(u_i), L_R) = \frac{Distance\ between\ points\ p(u_i)\ and\ L_R}{Z_R} \tag{39}$$

$$C_R = \frac{10}{N_T} * \sum_{j=0}^{N_T} C_j \tag{40}$$

$$C_j = min(\theta_i + d(p(u_i), L_R), i = 1,2, \ldots, N), \qquad j = 1,2, \ldots, N_T \tag{41}$$

Terrain collision detection is performed for every path segment to determine the terrain cost, *T*. For a specific point's latitude and longitude, the corresponding terrain height H_T is obtained. This value is subtracted from the height of the path at that point, H_P, to obtain the separation distance *h*. The cost is calculated based on the separation distance and user-defined safe flying altitude, which is set to 500 feet in this case. The terrain cost is given as follows:

$$h_i = H_{P_i} - H_{T_i} \tag{42}$$

$$T = \sum_{i=1}^{N} V_i, \quad V_i = \begin{cases} 1000, & h_i < 0 \\ \dfrac{500 - h_i}{50}, & 0 < h_i < 500 \\ 0, & h_i > 500 \end{cases} \tag{43}$$

The total fitness is found by combining the fuel efficiency, reconnaissance, and threat avoidance costs with user-defined weighting factors.

The capabilities of this path planner and the Virtual Battlespace make it possible to generate alternate paths in 3D and then display them using either 2D or 3D techniques. With many possibilities for information display in path planning, the first experiment was designed to better understand how to most effectively communicate path information to an operator. The second experiment then looked at the ability of an operator to re-task a UAV, utilizing the path-generating capabilities of the path planner and the same display techniques as in the first experiment.

Experiment 1 Design

The first experiment was conducted to determine the optimal means of displaying 3D path data for in-flight re-routing purposes. The scenarios presented to users assumed that unexpected circumstances had arisen, requiring the operator to pick an alternate path from the given choices. Such circumstances require an operator to accurately and rapidly assess the situation, interpret the alternatives, and make a decision. The first task involves the operator assessing the situation, which in this case is that the UAV needs to be re-routed, using one of the provided alternate paths. The second step (interpreting the alternatives) was of most interest: Which display method was most helpful in allowing users to accurately and quickly understand the relations between the presented paths and the original path? These tasks seem to require an understanding of the general shape of the paths and, for options close to one another, the ability to make precise distance judgments.

Experiment 1 Methods

Thirty-two (10 female, 22 male) undergraduate students were recruited from the Psychology department research participant pool. These students were all enrolled in an undergraduate psychology course. One student’s data was removed from the analysis due to a physical disability that limited her ability to use the controller to make rapid choices.

There were three groups in the first experiment. The first two groups were both using top-down 2D views: 2D numeric and 2D graphical. In the 2D numeric group, altitudes were displayed numerically next to each waypoint, as shown in Figure 66. In the 2D graphical group, altitudes were displayed using circles at each waypoint. The diameter of each circle indicated the altitude of the path at each point, such that one could think of the path sitting atop a sphere at that location. This final group was created in order to see if a novel graphical means might allow users to more-quickly assess the shape of a path. It is shown in Figure 67. For reference, the heights of the waypoints along the original path were also given using the same display techniques.

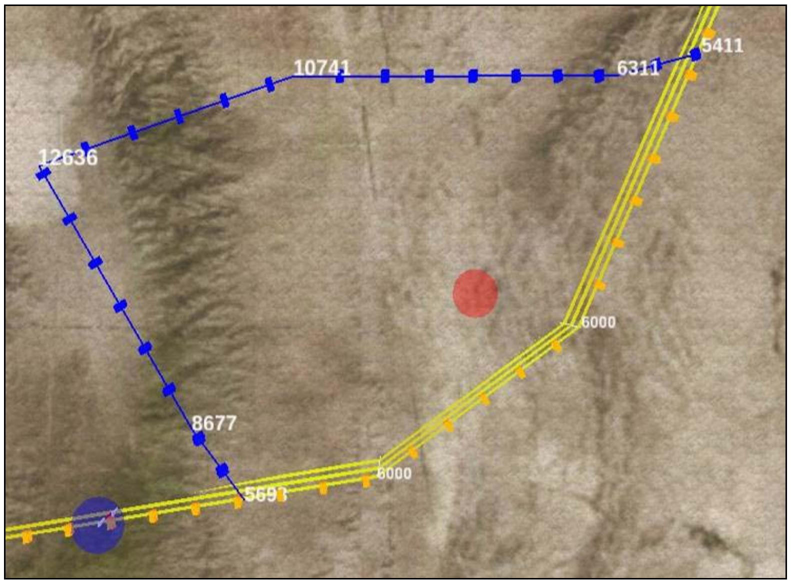


Figure 66. 2D numeric case

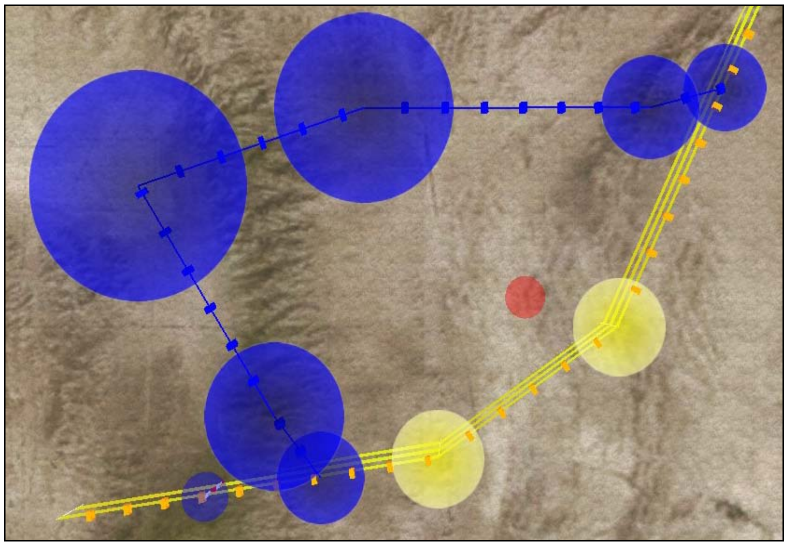


Figure 67. 2D graphical case

The third group was the 3D group. This group saw the paths depicted in 3D perspective views and had the ability to cycle through four pre-defined exocentric views, each displaying the battlefield from a different vantage point. Quaternion interpolation was used to pan between the view points. The four views are presented in Figure 68.

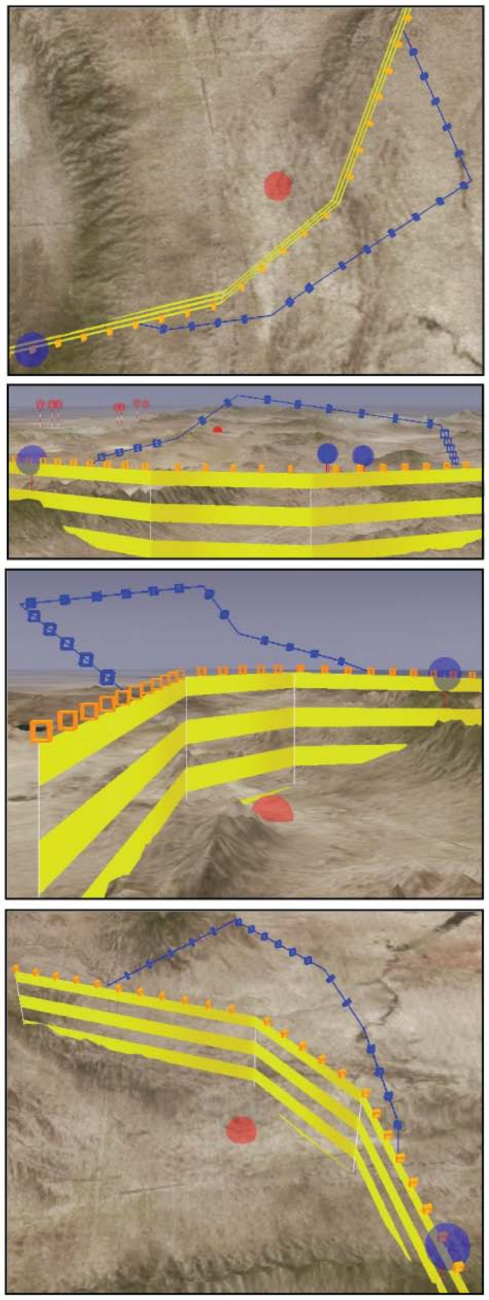


Figure 68. Four views for 3D case: top, side, isometric 1, and isometric 2

Participants in the 3D group were trained on how to use the gamepad controller to switch between views in the environment and all participants were briefed on how to complete the tasks (using the controller to switch between alternate paths and select one path). Part of the briefing was a demonstration by the experimenters of how to complete the task using a sample scenario very similar to the ones the users were about to encounter when they performed the task. They were told that, for unexplained reasons, in-flight re-routing was necessary. The objective was to choose the path that was closest in 3D space, on average, to the original (yellow) path. They were told that they were being timed but that there was no official time limit so they should concentrate on choosing the best path. All the actions of the users (such as switching between paths) were logged with the time they occurred.

Each user saw two scenarios. In both cases, an alert (see Figure 69) was displayed. When the user pressed the button to investigate the alert, the first alternate path was displayed and the timer started. Participants were given the ability to cycle forward and backward through the four path choices. The 3D group also had the ability to cycle forward and backward through the four pre-defined views. A short description of the mission briefing was included at the bottom of the screen at all times as a reminder for the user of the task objectives. That message is shown in Figure 70. All interaction between the user and the path planner was accomplished by pressing buttons on a gamepad controller.

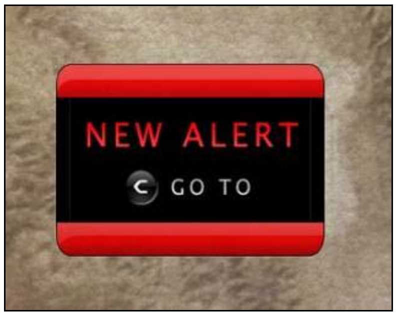


Figure 69. Alert for operator to re-plan path

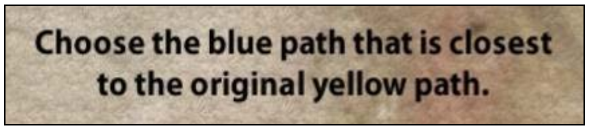

Choose the blue path that is closest to the original yellow path.

Figure 70. Mission objective reminder for user

Experiment 1 Results

As shown in Table 10, the average time to choose for either 2D group was much lower than that for the 3D group as the 3D group had more views to examine before making a choice. The accuracy of the participants in the 2D graphical group was on par with the other groups so the authors feel that it is a compelling possibility that this graphical representation may allow users to rapidly understand the general shape of a path. More research is needed on such graphical 2D representations.

Table 10. Average selection time by group

Group	Selection Time (s)
2D graphical	25.87
2D numeric	25.68
3D	35.81
All	29.12

In these trials, the 3D users qualitatively to take longer on average than the 2D users; however this difference was not quantitatively significant. Experimenter observations supported the belief that some of this time difference was due to the time required for cycling amongst the available 3D views.

Experiment 2 Design

A major objective of this research was to determine if a path-planning algorithm, coupled with a graphical interface, could enable inexperienced users to make re-routing decisions that were comparable to what an expert would decide, plotting the path manually. The second experiment was designed to investigate this question by providing the users with a more realistic path planning situation and comparing their choice to a path generated by an expert. The five path choices in each task were generated by the PSO path planning algorithm, using slightly different tradeoffs for the relative importance of avoiding the threat and staying close to the original path.

Experiment 2 Methods

The users in the 2D numeric group and the 3D group in experiment 1 participated in experiment 2. The users in the 2D graphical group were excluded because no good way was found to graphically display altitudes on the threat domes or no-fly zones. Since all users had already completed more basic tasks in experiment 1 they were not briefed again on how to use the controls to select a path. Threat domes and no-fly-zones were explained to the users.

Task 1

Participants were briefed on how to complete the first task. In this task they were told that unexpected threats are sometimes detected and in-flight re-routing is necessary. They were then taught how threat range is depicted in the Virtual Battlespace environment using a threat dome. Threat domes are depicted using topographic lines (red circles) that form a dome around the threat, as seen in Figure 71 for the 2D numeric case and Figure 72 for the 3D case.

Threats were assumed to have uniform range and effectiveness in all directions. Participants were told that the risk to the UAV was greater near the center of the dome, with outer regions of the dome being safer but still in range of the threat. They were told that they were being timed but that there was no time limit so they should concentrate on choosing the best path to avoid the threat while deviating as little as possible from the original path. The users were not told how to determine the relative importance of the two objectives included in the problem (threat avoidance and staying close to the original path).

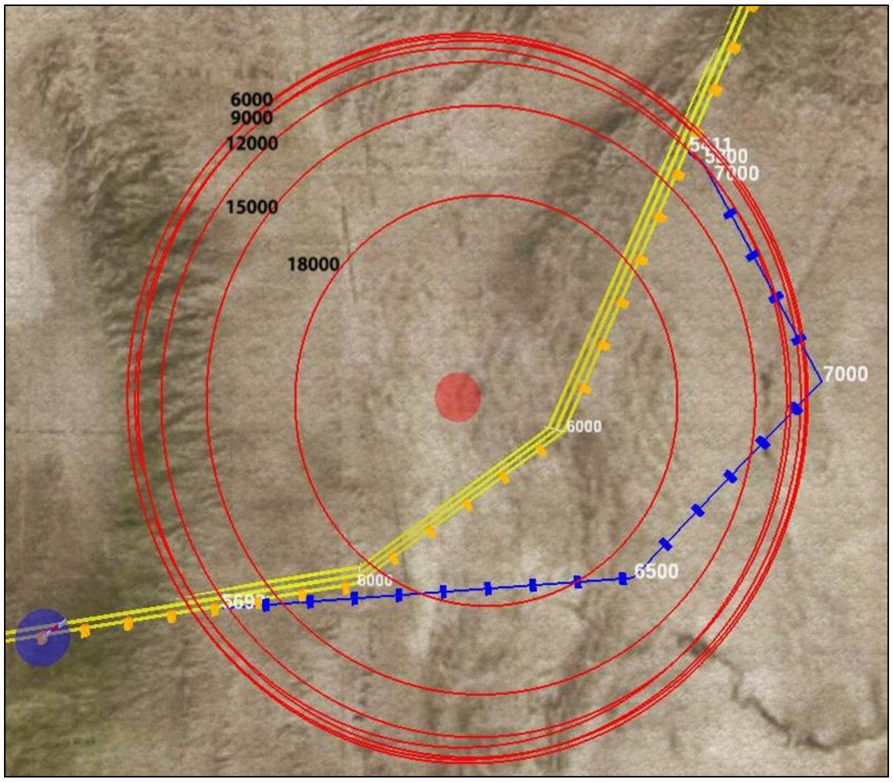
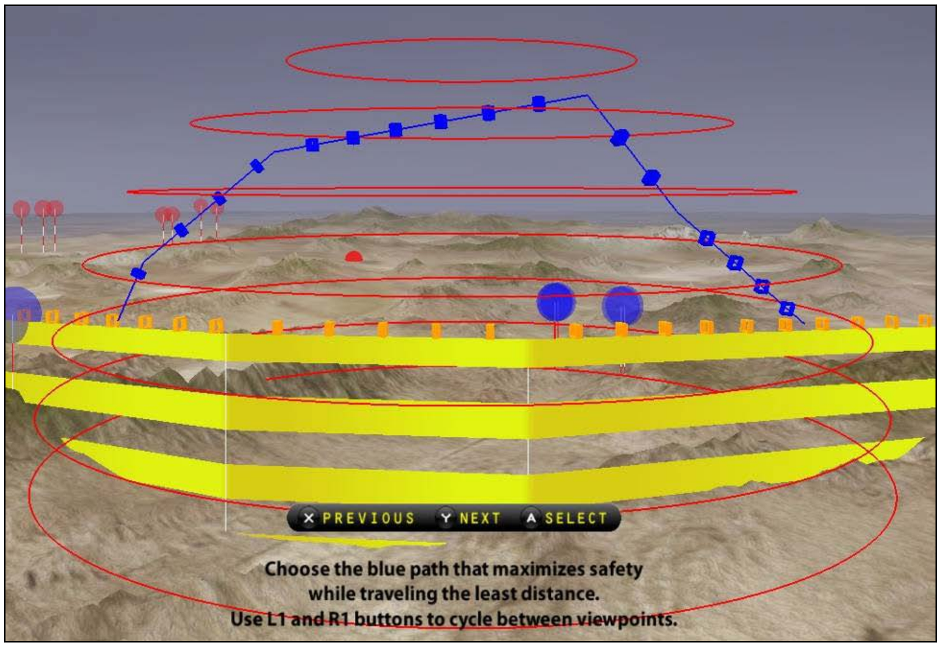


Figure 71. Threat dome representation, 2D numeric case



Choose the blue path that maximizes safety
while traveling the least distance.
Use L1 and R1 buttons to cycle between viewpoints.

Figure 72. Threat dome representation, 3D case

Task 2

In the second task, participants encountered a no-fly zone. They were told that a no-fly zone was similar to a threat dome, except that they absolutely could not enter a no-fly-zone (though entering a threat dome is acceptable). No-fly-zones were depicted with green lines to represent a ceiling above which a UAV should not fly. Figure 73 shows the no-fly zone in the 2D numeric case, where heights are given for the green contour lines. Figure 74 shows the no-fly zone from one of the 3D views.

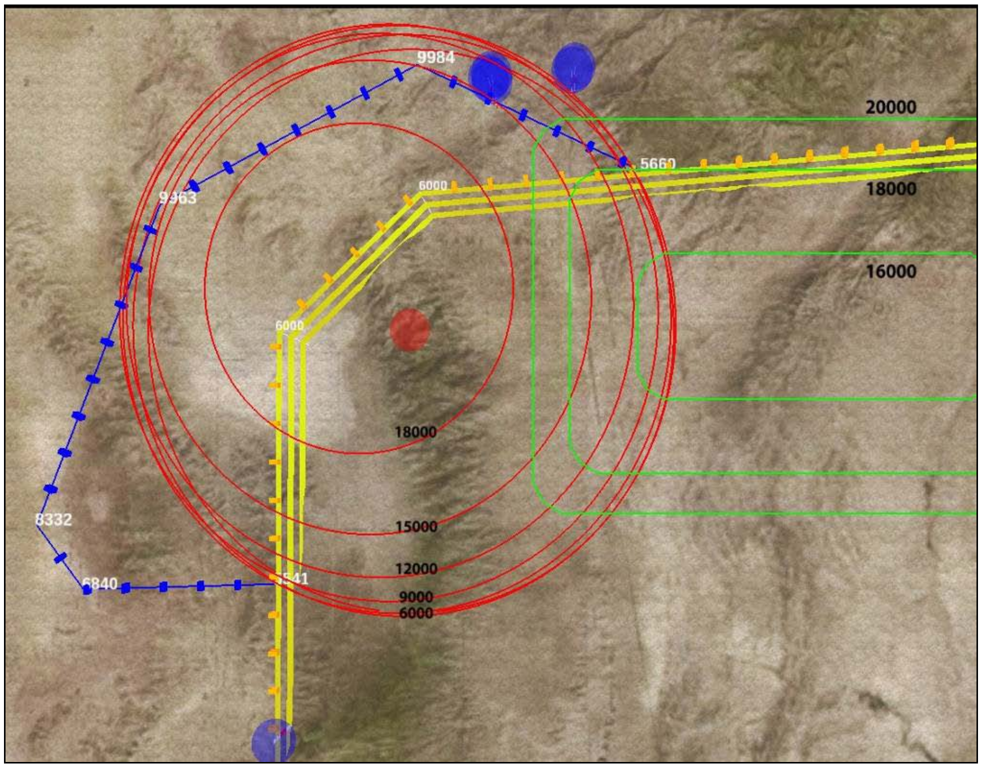


Figure 73. No-fly-zone representation, 2D numeric case

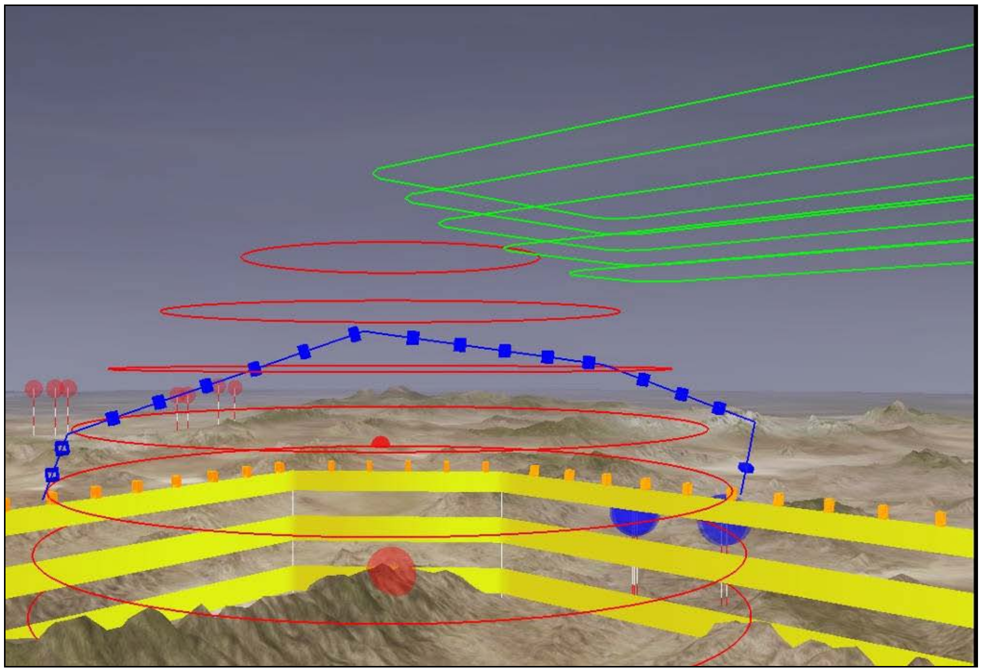


Figure 74. No-fly-zone representation, 3D case

Expert Path Creation

A manual path planner application was developed so that an expert could easily plot points and enter altitudes. This application depicted the scenario in a 2D top-down manner with numeric elevations at waypoints (the same view given to the 2D numeric group). The expert entered waypoints by clicking locations on the map and then manually entering altitudes in textboxes.

Experiment 2 Results

A quantitative and qualitative analysis was conducted for each task in experiment 2.

Task 1

In the first task, the users were attempting to balance avoiding the threat and staying close to the original path. How that tradeoff is balanced determines the path that is most appealing. If the desire to avoid the threat dominates, path 1 (which takes the long way around the threat) or path 0 (which travels high over the threat) might be good options. Path 3 which stays very close to the original path (taking it close by the threat) would only be selected by those willing to take large risks with the safety of the UAV. The other paths are more evenly balanced.

Fitness values were found for each of the alternate paths and the expert path using the PSO path planner cost function described earlier. Table 11 shows the fitness values for each of the objectives (fuel efficiency, reconnaissance, and threat avoidance) and the weighted total fitness.

Table 11. Cost values for paths for task 1

	Cost Values			
	Fuel	Recon	Threat	Total
Path 0	12.37	44.64	34.17	31.34
Path 1	15.36	67.69	29.78	35.65
Path 2	12.55	45.25	42.67	35.78
Path 3	11.72	20.09	78.38	47.14
Path 4	13.65	36.33	49.53	37.26
Expert	15.45	52.09	14.40	24.08

The expert path had relatively high values for fuel efficiency and reconnaissance, choosing instead to minimize threat avoidance. None of the paths presented to the users was as safe as the expert path, though path 1 had the most similar cost values. Path 1 goes around the opposite side of the threat to achieve a low threat avoidance cost, demonstrating the multi-modal nature of the path planning problem. Based on this comparison, 1 person in the 3D case and no one in the 2N case selected the correct path (path 1).

Similarity in cost does not necessarily mean the paths are similar in shape or heading. Therefore, a second metric was also used to determine the best path: separation distance in 3D space between the expert path and the alternate paths. To calculate this value, all the paths were broken into 200 segments of equal length. For each corresponding segment on the expert path and the alternate path, the vector distance was computed. The value presented in Table 12 is the average of those 200 separation distances for that alternate path. By this metric, path 0 is the best choice. Path 2 is a close second-best option (only 4 percent worse than the best option). By separation distance, 2 people in the 3D case and 1 person in the 2D graphical case selected the best path (path 0). Table 3 presents what the other users selected.

Table 12. Path separation and the number of users selecting each path for task 1

Path	**Average Distance from Expert (miles)**	**Selections** 2N	3D
0	1.16	1	2
2	1.22	2	2
4	1.35	6	3
3	1.62	1	2
1	3.79	0	1

In both groups, path 4 was chosen most often. This alternate path attempts to go around the threat without changing altitude significantly. The top-down and side views of these paths are plotted in Figure 75 and Figure 76.

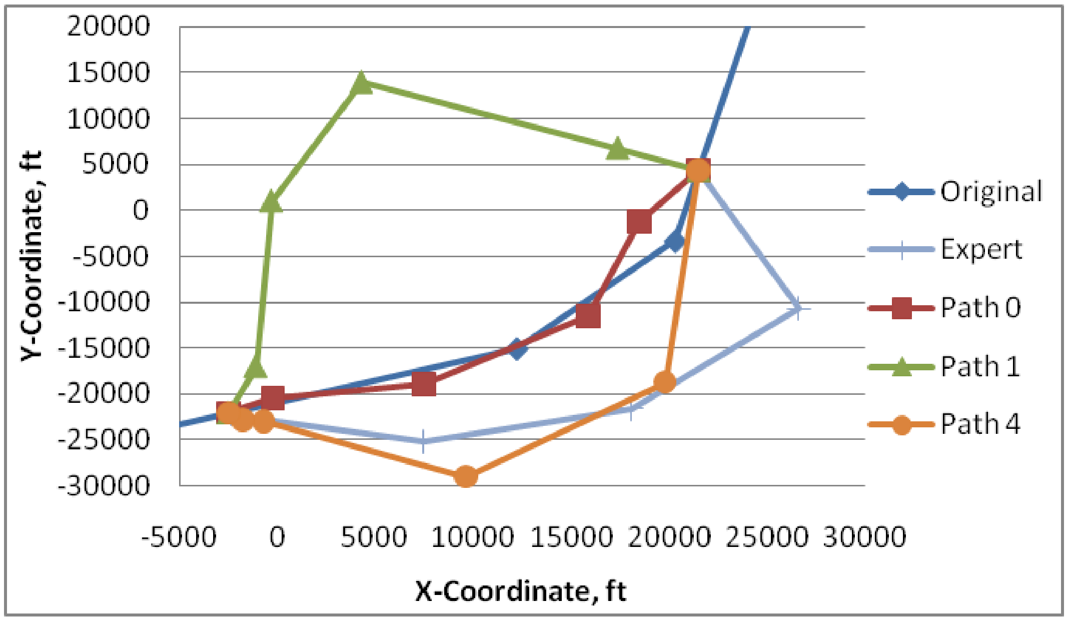


Figure 75. Top-down view of paths for task 1

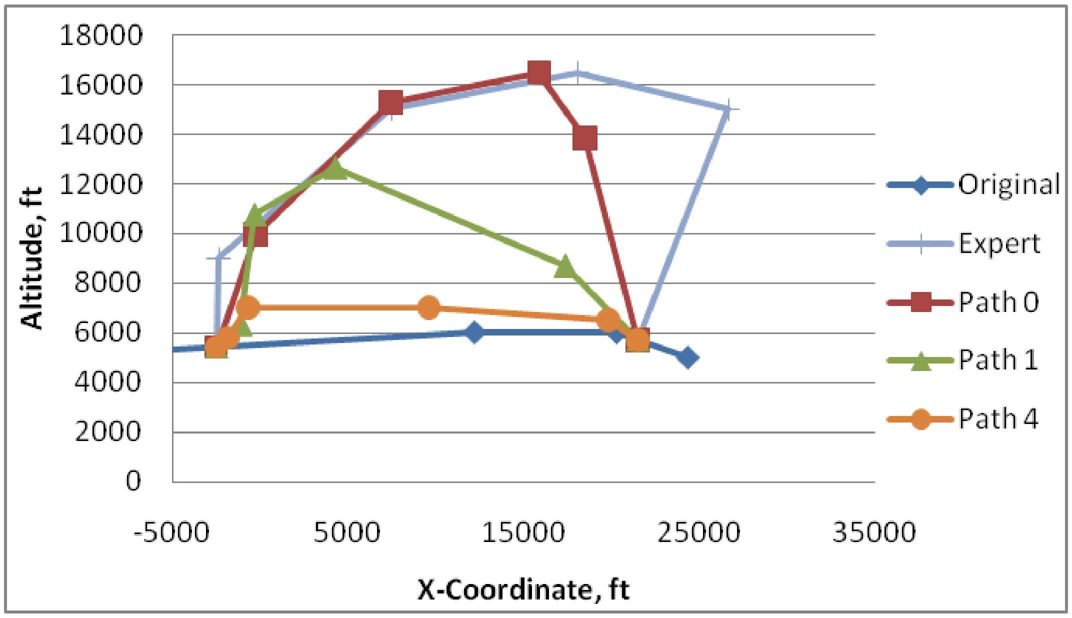


Figure 76. Side view of paths for task 1

Task 2

Task 2 of Experiment 2 was more complicated than Task 1 because of the addition of the no-fly-zone. Again, the users had to analyze the tradeoff between keeping the UAV safe and staying close to the original path. Because of the restriction that the path was to completely avoid the no-fly-zone, path 1 was the worst choice because it briefly entered the restricted area. One person from both the 2N case and the 3D case selected path 1. If the no-fly-zone was ignored, that path would be the safest alternative, so the users who selected it could have been trying to minimize time in the threat zone.

Paths 0, 3, and 4 are all similar in their threat avoidance costs: path 0 goes over the threat, path 3 takes the shorter way around the threat and path 4 takes the longer way around. Any of these would be good choices for the user who wanted to choose a safe path. Users who chose path 2 would be accepting more risk to stay closer to the original path. The actual cost values are presented in Table 13 for all the paths.

Table 13. Cost values for paths for task 2

	Cost Values			
	Fuel	Recon	Threat	Total
Path 0	11.20	80.44	19.57	32.69
Path 1	11.27	94.55	11.61	32.26
Path 2	13.19	35.91	29.20	26.88
Path 3	10.78	90.09	21.57	36.00
Path 4	16.39	51.43	22.94	28.42
Expert	13.88	61.26	12.58	25.07

As in Task 1, the separation distance between the expert path and each alternate path was calculated. Table 14 shows that, according to this metric, path 0 is the best choice. Path 2 was the second-best option. Figure 77 and Figure 78 present the original path, expert path, path 0, and path 2 for comparison purposes. It can be seen that the two alternate path choices lie on either side of the expert path, suggesting that the paths presented to the users are similar to the path that an expert would generate.

Table 14. Path separation and the number of users selecting each path for task 2

Path	Average Distance from Expert (miles)	Selections 2N	Selections 3D
0	1.12	2	4
2	1.24	3	4
3	1.88	1	1
1	2.21	1	1
4	2.56	3	0

Eight users selected path 0 or path 2 in the 3D case while only 5 did so in the 2N case. In both cases, this performance is better than that seen in task one. This leads to marginal significance that the 3D users were better than the 2D numeric users at selecting the correct path.

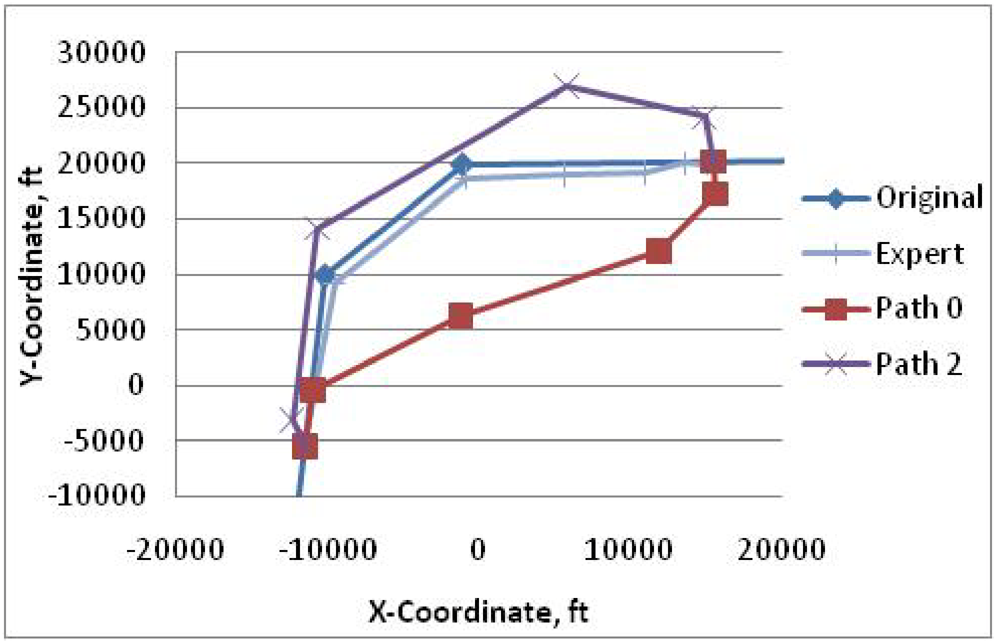


Figure 77. Top-down view of paths for task 2

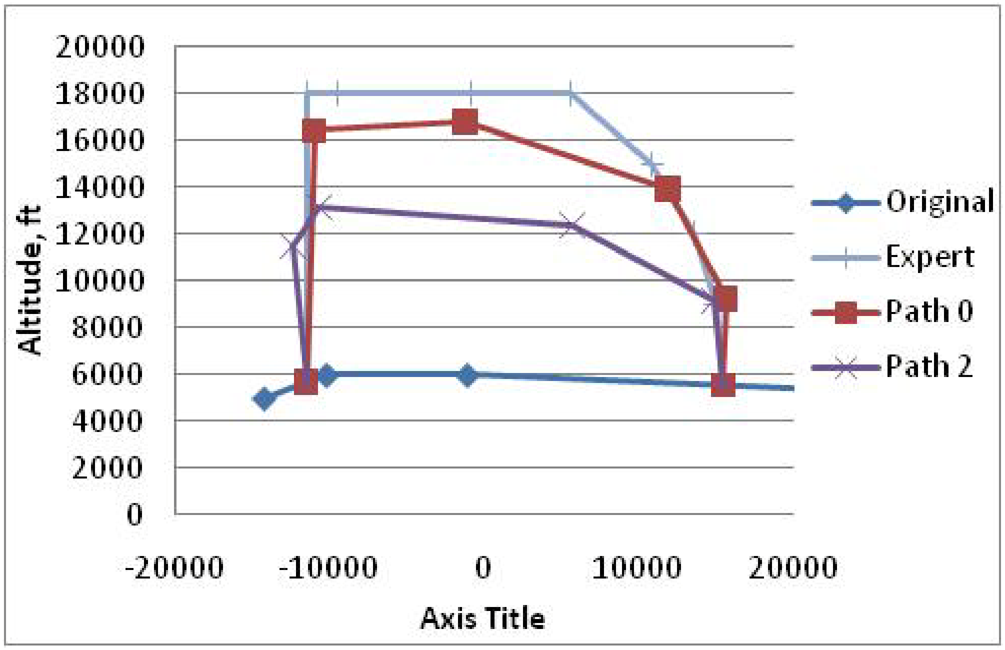


Figure 78. Side view of paths for task 2

Conclusion

In the first experiment, results suggest that each of the display methods is viable for depicting simple path alternatives. The results of the second experiment suggest that 3D visualization of alternate paths may be better in more complicated situations where the operator has a more complicated scene to understand before making a decision.

In both experiments, the users in the 3D case took longer to select a path than the users in the 2D cases. It is suspected that time spent switching between viewpoints to get different views of the paths creates this difference. Even with the increased time inherently required for 3D path visualization, enhanced accuracy may make it a good choice for many systems.

Future Work

More experiments are planned to study aspects of visualization and path planning in greater detail in the future based on the results of the study and input from experts in the field.

The users plan to repeat the first experiment with alternative paths where the separation distance from the original path is strictly controlled to see if the hardness of the task affects time and accuracy. Tests are also planned to see how users perform on these tasks when viewing the paths in stereo projection or when having freedom to navigate in the environment and view the paths from any position.

The second experiment will also be repeated with modifications to the visualizations of the no-fly-zone. Having more scenarios with a wider range of difficulties will provide a better gauge of how effective interactive path planning is at allowing novices to perform a more realistic set of planning tasks.

References

Barry, C.L., & Zimet, E. (2001, October). UCAVs Technological, Policy, and Operational Challenges, *Defense Horizons*, 3.

US Department of Defense (2007). Unmanned Systems Roadmap 2007-2032. Retrieved May 21, 2009, from http://auvac.org/research/publications/files/2007/unmanned_systems_roadmap_2007-2032.pdf

Kabamba, P.T., Meerkov, S.M., & Zeitz, F.H. (2006, March-April). Optimal Path Planning for Unmanned Combat Aerial Vehicles to Defeat Radar Tracking, *Journal of Guidance, Control, and Dynamics, 29*(2), 279-288.

Hasircioglu, I., Topcuoglu, H.R., & Ermis, M. (2008, July). 3-D Path Planning for the Navigation of Unmanned Aerial Vehicles by using Evolutionary Algorithms, *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, 1499-1506.

Kirkpatric, S., Gelatt, C.D., & Vecchi, M.P. (1983, May 13). Optimization by Simulated Annealing, *Science, New Series, 220*(5498), 671-680.

Mitchell, M. (1998). *An Introduction to Genetic Algorithms*, Cambridge, MA: MIT Press.

Eberhart, R.C., & Kennedy, J. (1995). A New Optimizer Using Particle Swarm Theory, *6th International Symposium on Micro Machine and Human Science*, 39-43.

Hassan, R., Cohanim, B., de Weck, O., & Venter, G. (2005, April 18-21). A Comparison of Particle Swarm Optimization and the Genetic Algorithm, *Proceedings of the 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*.

Thunholm, P. (2005). Under Time Pressure: An Attempt Toward a Prescriptive Model of Military Tactical Decision Making, *How Professionals Make Decisions*, Florence, KY: Routledge, 43-56.

Lurie, N.H., & Mason, C.H. (2007, January). Visual Representation: Implications for Decision Making, *Journal of Marketing, 71*, 160-177.

Shangming, W., Zefran, M., & DeCarlo, R.A. (2008, May). Optimal Control of Robotic Systems with Logical Constraints: Application to UAV Path Planning, *Proceedings of the IEEE International Conference on Robotics and Automation*, 176-181.

Soto, M., Nava, P.A., & Alvarado, L.E. (2007). Drone Formation Control System Real-Time Path Planning, *AIAA Infotech and Aerospace Conference and Exhibit*.

St. John, M., Cowen, M.B., Smallman, H.S., & Oonk, H.M. (2001). The Use of 2D and 3D Displays for Shape-Understanding versus Relative-Position Tasks, *Human Factors, 43*, 79-98.

St. John, M., Smallman, H.S., Bank, T.E., & Cowen, M.B. (2001). Tactical Routing Using Two-Dimensional and Three-Dimensional Views of Terrain, *Proceedings of the Human Factors and Ergonomics Society 45th Annual Meeting*, 1409-1413.

Smallman, H.S., St. John, M., Oonk, H.M., & Cowen, M.B. (2001, September-October). Information Availability in 2D and 3D Displays, *IEEE Computer Graphics and Applications,* 51-57.

Chapter 6 – Conclusions and Future Work

Conclusions

Several modifications were made to improve the performance of the PSO path planning algorithm developed for the Virtual Battlespace. The improved version of the PSO algorithm incorporates more accurate models for the reconnaissance and threat avoidance objectives and scales all the component costs to improve the correlation between the weights and the generated alternate paths. Using terrain information embedded in the Virtual Battlespace, the paths generated are feasible and safe from terrain interference. These improvements were noted in the different test cases used to evaluate the path planner. Most importantly, the paths were generated in real time to allow for efficient decision making by the UAV operator.

Meta-paths allow for effective visualization of multiple paths for decision-making purposes. The option of selecting one alternate path from a set of solutions ensures that the operator is involved in re-tasking the UAV. To facilitate the decision-making process, relative fitness and fitness component bars were generated for each path. The augmentation of the visualization with this information allows the operator to gain a better understanding of the tradeoffs made in selecting a particular path. These features of the path planner serve to enhance the operator’s ability to select (in an informed manner) the alternate path from the available options that will best meet the mission objectives.

Two alternate interaction techniques were developed that would allow the operator to more directly control the weights used by the algorithm in generating and ranking alternate paths.

Pre-optimization weight adjustment allowed the operator to select the weights that were used by the PSO algorithm and re-plan as many times as necessary. Post-optimization analysis allowed the operator to cull potential choices based on the relative importance of the objectives. Both methods show promise for more effectively incorporating operator preferences and mission objectives into the planning process.

The preliminary user study conducted on the path planner demonstrated the potential of the interface to allow novice users to re-task a UAV in a similar manner to what an expert would do. The feedback received from experts in the field of UAV control indicates that this is a relevant concept warranting further investigation.

Future Work

Multiple avenues are available for future improvements to this path planner. The first task would be to add dynamics to the model of the UAV. Currently, there are path planners that take into account vehicle dynamics such as maximum angle of ascent or descent and minimum radius of curvature to ensure the UAV is able to travel the alternate path [53, 54, 55]. Such improvements will be necessary to properly account for UAV's capabilities.

A second task would be to modify the path planner to be dynamic in nature by including time as a variable. There are a few path planners that allow threats to move and dynamically update the vehicle path to avoid them or model their locations probabilistically [56, 57]. In the current version of this path planner, all entities are considered stationary for the purpose of evaluating the cost function. An alternate path could easily be several minutes in length, meaning that entities could move several miles in that time and so the re-planned path may not properly avoid enemy entities. It should not be the entity location at the time path planning is performed, but the projected position of the entities at the time the UAV would be closest to them that are used in planning. Thus, forecasting would become an important part of this path planner.

A third modification would be to perform path smoothing during or after optimization. As seen in the results, it is possible for paths to make unnecessary loops to artificially improve their fitness value. The loops do not help the UAV accomplish its mission and could be eliminated by modifications to the cost function or a separate portion of the algorithm that would detect the loops and eliminate them.

A fourth modification would be to incorporate digital pheromones into the basic PSO algorithm for path planning [58]. This modification has shown performance improvements on test problems and would hopefully aid the path planner in finding better solutions or speed up the solution process.

A fifth task would be to incorporate line-of-sight calculations into the reconnaissance cost. Obstructions of the UAV’s view of the target created by mountains or man-made obstacles are not accounted for in the present model.

A sixth issue to investigate would be the number of path segments used in computing the path cost. The value needs to balance the computational cost of additional segments versus the better resolution those segments provide. A study where the number of segments was varied would help establish a reasonable baseline for how that value should be set to generate good alternate paths.

Finally, a sixth task would be to continue the user studies on the path planner. There is room to investigate topics ranging from new display techniques to understanding the risk operators will assume in the path planning process to assessing how well novice operators are able to re-task UAVs in simple and complicated situations.

Bibliography

[1] Barry, C. L., and Zimet, E., "UCAVs Technological, Policy, and Operational Challenges," *Defense Horizons*, Center for Technology and National Security Policy, National Defense University, No. 3, October 2001.

[2] Defense Industry Daily, "Air Force Requests $5.7B for 144 More Predators," http://www.defenseindustrydaily.com/images/AIR_UAV_MQ-1_Predator_lg.jpg [Retrieved 29 June 2009].

[3] UAV Forum, "X-45 UCAV – The Boeing Co.," http://www.uavforum.com/library/photo/x45.htm [Retrieved 29 June 2009].

[4] US Department of Defense, "Unmanned Systems Roadmap 2007-2032," http://auvac.org/research/publications/files/2007/unmanned_systems_roadmap_2007-2032.pdf [retrieved 21 May 2009].

[5] EDinformatics, "Predator Drones," http://www.edinformatics.com/math_science/robotics/predator.htm [Retrieved 29 June 2009].

[6] Liggett, K., "Multi-UAV Supervisory Control Interface Technology (MUSCIT)," http://cerici.org/documents/2007_workshop/5-1-AFRL-Patzek.pdf [retrieved 21 May 2009].

[7] Miller, C., "Trust in Adaptive Automation: The role of Etiquette in Tuning Trust via Analogic and Affective Method," *1st International Conference on Augmented Cognition*, 2004.

[8] FalconView, "Features of FalconView Open Source," http://www.falconview.org/trac/FalconView/wiki/OpenSourceFeatures [Retrieved 29 June 2009].

[9] Foo, J.L., Knutzon, J., Kalivarapu, V., Oliver, J., and Winer, E., "Three-Dimensional Path Planning of Unmanned Aerial Vehicle in a Virtual Battlespace using B-Splines and Particle Swarm Optimization," *Journal of Aerospace Computing, Information, and Communication*, Vol. 6, May 2009.

[10] Antoniou, A., and Lu, W., "Practical Optimization: Algorithms and Engineering Applications," Springer, New York, 2007.

[11] Lee, K.Y., and El-Sharkawi, M.A., "Modern Heuristic Optimization Techniques: Theory and Applications to Power Systems," Wiley-IEEE Press, February 2008.

[12] Mitchell, M., "An Introduction to Genetic Algorithms," MIT Press, 1998.

[13] Kirkpatric, S., Gelatt, C.D., and Vecchi, M.P., “Optimization by Simulated Annealing,” *Science, New Series*, Vol. 220, No. 5498, pp. 671-680, 13 May 1983.

[14] Eberhart, R.C., and Kennedy, J., “A New Optimizer Using Particle Swarm Theory,” *6th International Symposium on Micro Machine and Human Science, Institute of Electrical and Electronics Engineers*, Piscataway, NJ, pp. 39-43, 1995.

[15] Walter, B.E., Knutzon, J.S., Sannier, A.V., and Oliver, J.H., “Virtual UAV Ground Control Station,” *AIAA 3rd “Unmanned Unlimited” Technical Conference, Workshop and Exhibit*, Chicago, IL, September 2004.

[16] Ruff, H.A., Calhoun G.L., Draper, M.H., Fontejon, J.V., and Guilfoos, B.J., “Exploring automation issues in supervisory control of multiple UAVs,” *Human performance, situation awareness, and automation: Current research and trends*, Vol. 2, Lawrence Erlbaum Associates, Inc, Mahwah, NJ, pp. 218-222, 2004.

[17] Hassan, R., Cohanim, B., de Weck, O., and Venter, G., “A Comparison of Particle Swarm Optimization and the Genetic Algorithm,” *Proceedings of 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, Austin Texas, AIAA, 18-21 April 2005.

[18] Walter, B.E., Knutzon, J.S., Sannier, A.V., and Oliver, J.H., “Virtual UAV Ground Control Station,” *AIAA 3rd “Unmanned Unlimited” Technical Conference, Workshop and Exhibit*, 2004.

[19] Guang, Y., and Kapila, V., “Optimal path planning for unmanned air vehicles with kinematic and tactical constraints,” *Proceedings of the 41st IEEE Conference on Decision and Control*, Vol. 2, pp. 1301-1306, December 2002.

[20] Scherer, S., Singh, S., Chamberlain, L., and Saripalli, S., “Flying fast and low among obstacles,” *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2007)*, pp. 2023-2029, April 2007.

[21] Langelaan, J., and Rock, S., “Towards Autonomous UAV Flight in Forests,” *AIAA Guidance, Navigation, and Control Conference and Exhibit*, San Francisco, CA, Aug. 2005.

[22] Sanders, G. and Ray, T., “Optimal offline path planning of a fixed wing unmanned aerial vehicle (UAV) using an evolutionary algorithm,” *IEEE Congress on Evolutionary Computation*, pp. 4410-4416, September 2007.

[23] Nikolos, I.K., Zografos, E.S., and Brintaki, A.N., “UAV path planning using evolutionary algorithms,” *Innovations in Intelligent Machines – 1*, Springer, pp. 77-111, 2007.

[24] Kamrani, F. and Ayani, R., “Distributed Simulation and Real-Time Application,” *Proceedings of the 11th IEEE International Symposium on Distributed Simulation and Real-Time Applications*, pp. 167-174, October 2007.

[25] Jung, D., Ratti, J., and Tsiotras, P., “Real-time implementation and validation of a new hierarchical path planning scheme of UAVs via hardware-in-the-loop”, *Journal of Intelligent and Robotic Systems*, Vol. 54, No. 1-3, pp. 163-181, March 2009.

[26] Kabamba, P.T., Meerkov, S.M., and Zeitz, F.H., “Optimal path planning for unmanned combat aerial vehicles to defeat radar tracking,” *Journal of Guidance, Control, and Dynamics,* Vol. 29, No. 2, pp. 279-288, March-April 2006.

[27] Hasircioglu, I., Topcuoglu, H.R., and Ermis, M., “3-D path planning for the navigation of unmanned aerial vehicles by using evolutionary algorithms,” *Proceedings of the 10th Annual Conference on Genetic and Evolutionary Computation*, New York, pp. 1499-1506, July 2008.

[28] National Geophysical Data Center, “NGDC Tsunami Inundation Gridding Project,” http://www.ngdc.noaa.gov/mgg/inundation/tsunami/ [Retrieved 23 July 2008].

[29] Tenenbaum, D.E., Band, L.E., Kenworthy, S.T., and Tague, C.L., “Analysis of soil moisture patterns in forested and suburban catchments in Baltimore, Maryland, using high-resolution photogrammetric and LIDAR digital elevation datasets,” *Hydrological Processes,* Vol. 20, No. 2, pp. 219-240, 2006.

[30] Herrlich, M., “A Tool for Landscape Architecture Based on Computer Game Technology,” *17th International Conference on Artificial Reality and Telexistence,* pp. 264-268, 28-30 November 2007.

[31] Zhou, H., Sun, J., Turk, G., and Rehg, J.M., “Terrain Synthesis from Digital Elevation Models,” *IEEE Transactions on Visualization and Computer Graphics,* Vol. 13, no. 4, pp. 834-848, July-August 2007.

[32] Vandapel, N., Donamukkala, R.R., and Herbert, M., “Unmanned Ground Vehicle Navigation Using Aerial Ladar Data,” *The International Journal of Robotics Research*, Vol. 25, pp. 31-35, 2006.

[33] Virtual Terrain Project, “Digital Elevation Data,” http://www.vterrain.org/index.html [Retrieved 11 August 2008].

[34] National Geophysical Data Center, “The Global Land One-km Base Elevation (GLOBE) Project; A 30-arc-second (1-km) gridded, quality-controlled global Digital Elevation Model (DEM),” http://www.ngdc.noaa.gov/mgg/topo/globe.html [Retrieved 23 July 2008].

[35] Mittal, S. and Deb, K., “Three-Dimensional Offline Path Planning for UAVs Using Multiobjective Evolutionary Algorithms,” http://www.cse.iitk.ac.in/users/mshashi/papers/aiaa.pdf [Retrieved 29 May 2009].

[36] Sinopoli, B., Micheli, M., Donato, G., and Koo, T.J., “Vision Based Navigation for an Unmanned Aerial Vehicle,” *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1757-1765, May 2001.

[37] Thunholm, P., "Under Time Pressure: An Attempt Toward a Prescriptive Model of Military Tactical Decision Making," *How Professionals Make Decisions*, Routledge, Florence, KY, pp 43-56, 2005.

[38] Shrinivasan, Y., and van Wijk, J., "Supporting the Analytical Reasoning Process in Information Visualization," *Conference on Human Factors in Computing Systems*, New York, pp. 1237-1246, 2008.

[39] Lurie, N.H., and Mason, C.H., "Visual Representation: Implications for Decision Making," *Journal of Marketing*, Vol. 71, pp. 160-177, January 2007.

[40] Tominski, C., Schulze-Wollgast, P., and Schumann, H., "3D Information Visualization for Time Dependent Data on Maps," *9th International Conference on Information Visualization*, pp. 175-181, 2005.

[41] Estlin, T., Gaines, D., Chouinard, C., Fisher, F., Castano, R., Judd, M., Anderson, R.C., and Nesnas, I., "Enabling autonomous rover science through dynamic planning and scheduling," *IEEE Aerospace Conference*, pp. 385-396, 5-12 March 2005.

[42] Chandler, P.R., and Pachter, M., "Research issues in autonomous control of tactical UAVs," *American Control Conference*, 1998.

[43] Hocaoglu, C., and Sanderson, A.C., "Planning Multiple Paths with Evolutionary Speciation," *IEEE Transaction on Evolutionary Computation*, pp. 169-191, June 2001.

[44] Shangming W., Zefran, M., and DeCarlo, R.A., "Optimal control of robotic systems with logical constraints: Application to UAV path planning," *Proceedings of the IEEE International Conference on Robotics and Automation*, pp.176-181, May 2008.

[45] Ceccarelli, N., Enright, J.J., Frazzoli, E., Rasmussen, S.J., and Schumacher, C.J., "Micro UAV Path Planning for Reconnaissance in Wind," *American Control Conference*, pp.5310-5315, July 2007.

[46] Hall, R., "Path Planning and Autonomous Navigation for use in Computer Generated Forces," *Combat Simulations, FLSC*, June 2007.

[47] Soto M., Nava, P.A., and Alvarado, L.E., "Drone Formation Control System Real-Time Path Planning," *AIAA Infotech and Aerospace Conference and Exhibit*, May 2007.

[48] Kwangjin, Y., and Sukkarieh, S., "3D smooth path planning for a UAV in cluttered natural environments," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.794-800, September 2008.

[49] Ruz, J.J., Arevalo, O., Pajares, G., and de la Cruz, J.M., "Decision Making among Alternative Routes for UAVs in Dynamic Environments," *IEEE Conference on Emerging Technologies and Factory Automation*, pp. 997-1004, September 2007.

[50] Foo, J.L., Knutzon, J.S., Oliver, J.H., and Winer, E.H., "Three Dimensional Path Planning of Unmanned Aerial Vehicles using Particle Swarm Optimization,"

Proceedings of the 11th AIAA/ISSMO multidisciplinary analysis and optimization conference, Portsmouth, VA, 2006.

[51] Malhotra, A., Oliver, J. H., and Tu, W., "Synthesis of Spatially and Intrinsically Constrained Curves Using Simulated Annealing," *Journal of Mechanical Design*, March 1996.

[52] Piegl, L. A., and Tiller, W., "B-Spline Curves and Surfaces," *The NURBS Book*, 2nd ed., Spring-Verlag, New York, pp. 81–116, 1997.

[53] Anderson, E.P., Beard, R.W., and McLain, T.W., "Real-time dynamic trajectory smoothing for unmanned air vehicles," *IEEE Transactions on Control Systems Technology*, Vol. 13, No. 3, pp 471-447, May 2005.

[54] McLain, T.W., and Beard, R.W., "Coordination Variables, Coordination Functions, and Cooperative-Timing Missions," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 1, January-February 2005.

[55] Zheng, C., Xu, F., Hu, X., Sun, F., and Yan, P., "Online Route Planner for Unmanned Air Vehicle Navigation in Unknown Battlefield Environment," IMACS Multiconference on Computational Engineering in Systems Applications, Vol. 1, pp. 814-818, 4-6 October 2006.

[56] Kim, Y., Gu, D., and Postlethwaite, I., "Real-time path planning with limited information for autonomous unmanned air vehicles," *Automatica*, Vol. 10, pp. 696-712, 23 July 2007.

[57] Rathbun, D., Kragelund, S, and Pongpunwattana, A., "An Evolutionary Based Path Planning Algorithm for Autonomous Motion of a UAV Through Uncertain Environments," *Proceedings of the AIAA Digital Avionics Systems Conference*, 2002.

[58] Kalivarapu, V., Foo, J., and Winer, E., "Improving solution characteristics of particle swarm optimization using digital pheromones," Structural and Multidisciplinary Optimization, Vol. 37, No. 4, pp. 415-427, January 2009.

Acknowledgements

I would like to thank those who have helped me through the last two years as I have been working on my master's degree. First, I give praise to Jesus Christ, my Lord and Savior, who has provided me with both the ability and the perseverance to complete this work. Thank you to my parents Keith and Sandy and my brother, Tyler, and sister, Allison. They have been a continual source of encouragement to me especially when research was the most time-consuming and tiring. I am also grateful to Lindsey, my fiancé, who has supported me throughout the writing of my thesis. Her willingness to allow me to work long hours and her help in editing my thesis have been greatly appreciated. Thank you to all my family and friends for the joy you have brought to the last two years.

I would extend a sincere thank you to Eliot Winer, my major professor, for being willing to take me on at the beginning of my master's program and for guiding me along the way. I would also thank James Oliver and Stephen Gilbert for being on my committee and encouraging my research. Thank you to Eric Foo for overseeing my work and answering my questions. Finally, I need to thank all the members of my research group who have helped make this a great experience, especially Joanna Peddicord who spent many hours running test scenarios for me.