12-11-2017

# Autonomous UAV Path Planning for Wildfire Data Collection

John Bent

# AUTONOMOUS UAV PATH PLANNING FOR WILDFIRE DATA COLLECTION

by

JOHN BENT

Under the Direction of Xiaolin Hu, PhD

## ABSTRACT

Teams of unmanned aerial vehicles (UAVs) can be employed in the real-time collection of data for dynamic data-driven simulations of wildfires to provide more accurate simulation predictions. The goal of this thesis was to identify and simulate path planning algorithms and inter-UAV negotiation protocols that best govern and divide responsibility among a team of autonomous UAVs continuously observing the perimeter of a wildfire containing sections of varying importance – ranging from low-value to high-value target areas. Several benchmark algorithms were implemented as well to measure the effectiveness of the new algorithms. The performance of each algorithm was measured by simulating a team with varying number of UAVs moving around the perimeter of multiple wildfires under differing conditions. The newly developed path planning and negotiation algorithms were found to increase the effectiveness of the team over the case of the benchmark algorithms. The increase in efficiency can be attributed to the UAVs' more selective choice of destination and the equal division of responsibility between UAVs.

INDEX WORDS: UAV, Drone, Autonomous path planning, Continuous perimeter sweeping, Wildfire, Dynamic data-driven simulation

# AUTONOMOUS UAV PATH PLANNING FOR WILDFIRE DATA COLLECTION

by

JOHN BENT

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science

in the College of Arts and Sciences

Georgia State University

2017

# AUTONOMOUS UAV PATH PLANNING FOR WILDFIRE DATA COLLECTION

by

JOHN BENT

## DEDICATION

This thesis is dedicated to my family, who has been a constant source of support for me over the years.

## ACKNOWLEDGEMENTS

I would like to thank my committee members for lending their expertise and their time. A special thanks to my committee chairman, Dr. Xiaolin Hu, who gave me years of guidance and encouragement and assisted me with this project. You are one of the best professors I have ever had, and I could not have asked for a better advisor. Thank you, Dr. Ashwin Ashok for sparking my interest in the field of Computer Vision, teaching for the first time an excellent course on the subject. I consider it one of the most valuable experiences I have had at Georgia State.

# TABLE OF CONTENTS

# LIST OF TABLES

## LIST OF FIGURES

## LIST OF ABBREVIATIONS

| | |
|---|---|
| DEVS | Discrete Event System Specification |
| UAV | Unmanned Aerial Vehicle |
| ROS | Rate of Spread |
| HITL | Hardware-in-the-Loop |
| COI | Center of importance |

# 1 MOTIVATION

The frequency and duration of wildfires in the United States underwent a sharp increase in the mid-1980's due to changes in the climate, such as warmer temperatures and lower levels of precipitation [1]. While the frequency and duration of wildfires have remained at this increased level, the total area of destruction has maintained an upward trend for the past three decades [2]. The annual growth in wildfire-burned area suggests that the current efforts to prevent and suppress wildfires are not keeping pace with the growing threat.

## 1.1 Wildfire Simulations

Simulation of wildfires is a burgeoning area of research with the goal of predicting wildfire behavior across time and space. These simulations can be used proactively in understanding the causes and the dynamics of wildfires so that forest managers can take steps to lessen the potential damage from wildfires or prevent them altogether. Simulations can also aid in constructing strategies for suppression of an active wildfire. Firefighting managers are often faced with short timeframes in which to make decisions about the deployment of limited firefighting resources to a dynamically changing fire. Wildfire simulations offer probabilistic guidance to inform the experienced firefighting manager's intuition [3].

The wildfire simulation is a valuable tool in active fire scenarios because of the high degree of uncertainty surrounding wildfire behavior; however, this uncertainty also limits the accuracy of the simulations, which are based on best estimations of stochastic factors like wind and precipitation and are only approximations of the real behavior of the fire. Wildfires have been known to create their own weather conditions and exhibit dynamic behaviors like fire whirls, which are vortexes of hot air that can transport burning material well beyond the fire front and over manmade clearings or roads [4]. Therefore, as the simulation progresses further into the future, the predicted state of the wildfire in the simulation will deviate further from the actual state of the wildfire.

## 1.2 Dynamic Data Driven Simulation

In [5], Hu presents *dynamic data driven simulation* as a new simulation paradigm that offers a solution to the problem of a simulation's deviation from the real system. Dynamic data driven simulations rely on the idea of *data assimilation*, or the incorporation of observed data from the real system into a running simulation model, coupling the simulation system with the real system. The assimilation of real-time data into a wildfire simulation improves the results of the simulation by providing a more accurate starting state of the fire.

The benefit derived from data assimilation is dependent not only on the observed data itself, but also on information about the observations, or the *metadata*, which can include the location, time and frequency of the observations. The values of these attributes are of critical importance to the quality of a dynamic data driven simulation and should be optimized to gain the greatest amount of information from the real system. In other words, there are certain observations that are more valuable than others. With limited resources for data collection, a methodology should be in place to best utilize those resources by seeking out the highest-value data.

In the case of wildfire systems, particular areas of the fire behave with more uncertainty than others, and that uncertainty is correlated with the *rate of spread* in that part of the fire. For example, the fire front, where the fire is spreading fastest, typically downwind, can develop *spotting* when a fire whirl picks up burning material and sends it thousands of meters beyond the fire front, igniting a new, smaller fire. This type of behavior and the high rate of spread itself lead to higher uncertainty in the location or state of the fire front. The rear of the fire is usually spreading the slowest because wind is blowing from the unburned area towards the burning or already-burned area; thus, the certainty of the fire’s behavior here is comparatively high because it is evolving slower than on the front. The implication of varied uncertainty is that a simulated wildfire, which is attempting to model this real system, is more likely to deviate in the areas with higher rates of spread; therefore, observations for data assimilation should be concentrated in those areas with higher rates of spread, and the frequency of said observations should be directly proportional to the rate of spread. The long-term goal of this research is to formalize a data collection methodology for dynamic data driven simulations so that simulation designers can quickly realize a data collection technique for a new system.

## 1.3 Unmanned Aerial Vehicles

Some methods of data collection are better suited for wildfire simulation than others due to the non-stationary, dynamic nature of the wildfire system. One of the simplest approaches would be to place temperature sensors on the ground surrounding the active wildfire. The sensors would transmit temperature readings to a host for data assimilation into a running simulation. Ground sensors are relatively inexpensive and can be distributed widely for simultaneous measurements at many parts of the fire. The biggest downside to this method is that it suffers from the same uncertainty that the simulation does; the strategy for deployment of ground sensors would need to consider where the areas of highest rate of spread would be and deploy the greatest concentration of sensors in that area. However, the behavior of the fire at that future state cannot be known at the time the sensors are deployed, and the repositioning of ground sensors in an active wildfire environment would be both time consuming and dangerous.

Another approach is the use of manned aerial vehicles to optically observe the fire from the sky. A pilot or pilots would navigate aircraft equipped with visual or infrared sensors over the fire. A team of pilots would be able to communicate and effectively share the responsibility for observing the fire. The main benefit of manned aircraft over ground sensors is the ability of a pilot to change course as needed to observe sections of the wildfire with a frequency appropriate to their importance. The obvious downside is that piloted aircraft are expensive to operate and carry the added risk of loss of human life.

There is a wide disparity in the strengths and weaknesses of ground sensors and manned aerial vehicles as methods for wildfire data collection. Recently, unmanned aerial vehicle (UAV) technology has matured to a point that makes it an ideal choice for wildfire data collection. UAVs combine the best qualities of ground sensors and manned aircraft while providing additional benefits. Most models of UAVs are now inexpensive enough that the cost of purchase and operation of a team of UAVs is less than leasing a piloted aircraft for a dangerous mission. A single UAV is still more expensive than a single ground sensor, but it is affordable enough to purchase and operate a team; UAVs are also able to move continuously, giving an obvious advantage over the ground sensor. UAVs can stay in the air for up to 30 hours and do not need the levels of visibility that a pilot would need, allowing them to continue through smoky conditions at any time of day or night. Finally, UAVs would eliminate the risk to human life that both ground sensor deployment and manually operated aircraft entail [6].

One important characteristic of UAVs is that they can be programmed to autonomously carry out a mission, either as an individual or by communicating with a team of multiple UAVs. This can potentially be another advantage over piloted aircraft, although the algorithms guiding the UAV(s) would need to outperform the pilot or team of pilots, in terms of the quality of data collection. The goal of this thesis is to develop the algorithms that will guide an autonomous team of multiple UAVs in collecting data from a wildfire with the purpose of assimilating the most valuable data into a dynamic data driven wildfire simulation.

## 1.4 Path Planning

As mentioned in section 1.2, observation of a section of a wildfire with a higher rate of spread is more valuable than an observation of a section with a slow rate of spread, and those sections that are spreading faster should be observed at a higher frequency than those spreading slowly. This is the guiding principle in developing a path planning algorithm for a UAV. The optimal route is the one that allows the UAV to observe the most valuable data in the least amount of time. The value, or *importance*, of a section of wildfire is determined by its rate of spread and the time since that section was last observed, with sections having been observed most recently being less important than sections that have gone unobserved for a longer interval. The time since the last observation must be included in determining importance so that the UAV will observe sections with low rates of spread, albeit with a lower frequency than those with high rates of spread. Without considering this time interval, the UAV would only deem the section with the highest rate of spread an appropriate target, never observing other slower moving sections. Because the time interval is involved in calculating the importance of a section of fire, the optimal route can also be defined as the route that minimizes the total importance of all sections of the fire.

## 1.5 Negotiation

When more than one UAV is involved in the observation of the wildfire, the data collection process can be further optimized by strategically dividing responsibility for observing regions of the fire. A naïve approach would be to divide the wildfire into equal sections for each UAV to observe, however, this method neglects to consider the varying importance of different sections of the fire and is not able to handle a dynamically changing system. An equally divided fire would assign regions with widely varying importance to the UAVs, so the region with the higher importance is being observed just as frequently as the region with lower importance. A more effective approach would be to divide the wildfire into regions of size inversely proportional to their importance. This way, each UAV would be responsible for observing regions with equal, or at least similar, levels of importance, which is in accordance with the goals of effective data assimilation.

Because the wildfire is dynamically changing, the region for which each UAV is responsible may grow or shrink in size and importance over time. It is important that any arising disparities in the responsibility between UAVs be flattened out. This can be achieved by implementing a negotiation protocol, wherein the UAVs periodically initiate a communication process with the goal of finding out how important their respective regions are and negotiating the transfer of subregions between UAVs with imbalanced responsibilities in an effort to maintain a fair and more effective data observation load among the team members.

## 1.6 Objective

The overarching objective of this thesis is to develop a data collection strategy using UAVs for efficiently collecting the highest value data from a wildfire in an effort to maximize the accuracy of dynamic data driven wildfire simulations. The broader objective is broken down into two distinct but interrelated tasks: developing the most effective path planning algorithm for a UAV observing a wildfire and developing the most effective negotiation algorithm for a team of UAVs observing a wildfire.

The algorithms are implemented using the DEVSJAVA simulation environment with an agent-based UAV model integrated into the DEVS-FIRE wildfire simulation developed by Hu.

The distinction should be made that the simulations developed for this thesis are not dynamic data driven simulations, but rather discrete event simulations that model the behavior of UAVs gathering data around an active wildfire for assimilation into a *hypothetical* dynamic data driven simulation.

The UAV teams are simulated using different combinations of path planning and negotiation strategies around a variety of wildfires. The resulting importance values for each scenario are collected and analyzed to better understand the UAV team's behavior and to determine how well the algorithms performed under certain conditions.

### 1.7 Organization of Thesis

The rest of this paper is organized as follows. Section 2 provides some background to this research, including studies that inspired the initial algorithms and the preliminary implementation using the NetLogo simulation environment. Section 3 gives a formal definition of the problem. A cost function is defined to put the problem in a mathematical context. The path planning and negotiation algorithms are detailed in section 4. Section 5 lays out the simulated experiments and gives an analysis of the results individually. Section 6 draws conclusions from the study as a whole and mentions future work that would be valuable should the research continue.

# 2 BACKGROUND

## 2.1 Related work

The path planning algorithms developed for this thesis are inspired by the work of Ahmadi et. al. [7], who developed the idea of continuous area sweeping tasks, in which a single autonomous robot performs routine surveillance of an area with subareas of varying degrees of importance. Rather than continuous area sweeping, this thesis focuses on continuous *perimeter* sweeping. The work also introduces a cost function that guides the robot towards areas of higher importance more frequently, which is used as a model for the cost function in this thesis. The problems are similar, but the transformation from a 2D problem into a 1D problem brings about some subtle differences.

That research was continued by Ahmadi et. al. [8] to expand the single-robot case into a multi-robot system for continuous area sweeping. The multi-robot system employs the same path planning formulation as the single-robot system, but it also relies on a negotiation procedure to divide the area into subareas to become the responsibility of each individual robot and dynamically swap parts of those subareas between robots. The negotiation procedure in [8] involves simulating the robot system to calculate which robot will be able to reach a specific area first; the simulation results are the basis on which the negotiation of responsibility is performed. This high degree of precision for the negotiation procedure is unnecessary in the case of continuous perimeter sweeping, so, instead, a less computationally expensive method is developed for this study.

## 2.2 NetLogo Simulation

The path planning and negotiation algorithms developed for this thesis were originally implemented as an agent-based simulation model using the NetLogo simulation environment. This original implementation considered the problem of continuous *area* sweeping; that is, the UAV team was modeled to monitor a 2D area of the wildfire rather than the 1D problem of monitoring the edge or perimeter of the wildfire. The mathematical wildfire model was greatly simplified in the NetLogo implementation, but served as a suitable testbed for the algorithm development. The DEVSJAVA implementation detailed in this thesis was directly motivated by the work done for the NetLogo simulation and began as a 1D projection of the 2D solution developed in NetLogo.

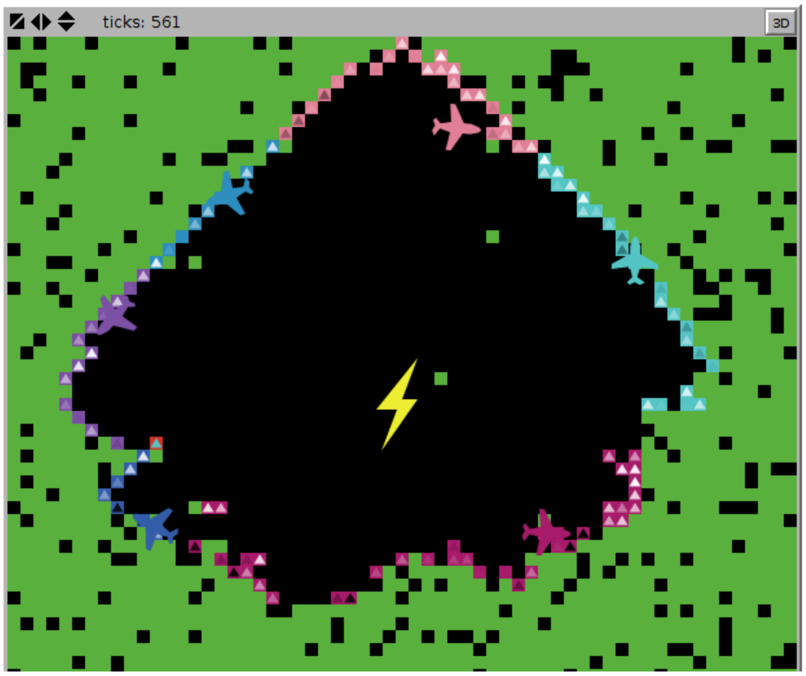


*Figure 2.1 Graphical display of NetLogo implementation*

## 2.3 Wildfire Model

This thesis involves the design of an agent-based UAV simulation model implemented using DEVSJAVA that interacts with the DEVS-FIRE simulation environment created by Ntaimo and Hu [9] [10]. DEVS-FIRE is based on discrete event system specification (DEVS) and employs a cellular space model for wildfire spread. Data for terrain, fuel, and weather are incorporated into the cellular space; each cell in a 2D grid has its own attribute values that fit the observed conditions of the real system at that location. These attributes determine the *maximum rate of spread* in any given cell, and this value is used in calculating the importance of the cell for the UAV model. The cells in the grid are coupled together and, when a cell is ignited, Rothermel's widely-used mathematical model for wildfire spread is used to calculate the fire spread within that single cell. Once the fire spread in a cell reaches a certain threshold, it spreads beyond that cell, and one or more neighboring cells become ignited.

A firefighting-agent model, which is intended to simulate an agent moving in a direction parallel to the perimeter of the fire, was developed in [9] and forms the basis for the implementation of UAV movement in this thesis. Specifically, the UAV model employs the same strategy for locating cells on the perimeter of the wildfire, although some minor improvements were made to this procedure.

# 3 PROBLEM DEFINITION

As mentioned previously, the broad objective of this work is to define behavior in autonomous UAV teams that guides the UAVs to visit and collect data from a wildfire perimeter most effectively for assimilation into a dynamic data driven simulation. The most effective data collection technique can be stated as the routing and cooperative behavior of a UAV team that allows the UAVs to gather the most important data in the smallest amount of time and to do so in a continuous procedure that emphasizes data collection frequency as a function of the importance of that data. This description of effective behavior gives an intuitive understanding of the problem, but a more formal definition is needed to construct the algorithms that elicit this behavior.

## 3.1 UAV region

The UAV is responsible for visiting “sections” of the wildfire perimeter for which that UAV is responsible. In the DEVS-FIRE simulation environment these sections are best thought of as the cells in the 2D cellular space. Because the UAVs are only responsible for observing the wildfire perimeter, the cells that the UAVs are responsible for are those that are in a “burning” state and neighboring at least two cells in an “unburned” state; even though the wildfire simulation may have some depth, or consist of a wide swath of “burning” cells, the UAVs should only visit those cells on the extreme perimeter of the wildfire. The perimeter forms a continuous 2D path outlining the shape of the wildfire. Figure 3.1 shows a DEVS-FIRE simulation with red “burning” cells, black “ burned” cells, green “unburned” cells, and a perimeter of multi- colored “burning” cells that the UAVs are responsible for visiting.

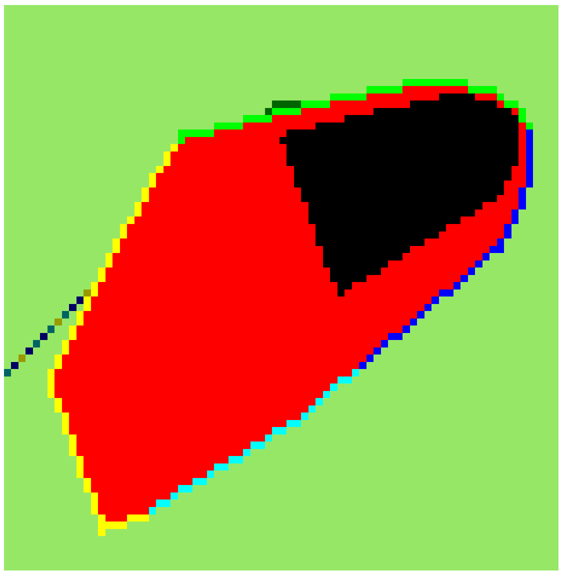


*Figure 3.1 Graphical display of DEVSJAVA UAV simulation with 4 different-colored UAV regions depicted*

Each UAV in the team takes responsibility for a single contiguous “region” of cells from the perimeter (each region having a different color in Figure 3.1). The cells within each UAV’s region must only be neighboring other cells of that region, except for two cells - one at each of the opposing extremes of the region - which may also neighbor one cell from the neighboring UAV’s region. This definition of a UAV’s “region” has important implications in the negotiation procedure. UAVs are only allowed to negotiate with neighboring UAVs, or UAVs with regions that contain a cell neighboring a cell in that UAV’s region, and the negotiation only allows the transfer of contiguous subregions beginning with that neighboring cell. This constraint ensures that each UAV’s region will remain intact as one contiguous chunk of cells after negotiation.

## 3.2 Cell importance

A cell’s “importance” value is proportional the fire’s rate of spread within that cell and the time since a UAV has last visited the cell. The rate of spread (ROS) is determined by multiple factors, including the amount of available fuel in the cell, the slope of the cell’s terrain, and the weather at the cell. ROS is an indicator of the uncertainty in the location of a section of wildfire; the location of a faster moving part of the fire is more likely to deviate from the prediction quicker than the slow moving parts. The uncertainty, and therefore ROS, are central in determining how frequently data should be collected from a particular area. The time since the cell was last visited is defined as the difference between the current simulation time and the time since a UAV entered, or “visited”, that cell. The UAVs move between cells in a discrete event fashion. When a UAV enters a cell it calculates the time until it will reach the center of the next cell and schedules an event at that time. After the scheduled time elapses the UAV enters the next cell, and this cell is considered to have been visited at that exact time. Likewise, the location of the UAV at that time is approximated as the center of this cell. Cells become more important the longer they are left unobserved because the likelihood that the real location of that part of the fire has deviated from the simulation’s prediction grows with each time step.

While the importance value for a cell is proportional to the ROS and the time since last visit, the degree to which it is proportional is somewhat arbitrary and may be determined by the needs of the user. For this thesis, two definitions of importance will be used, each proportional to ROS and time since last visit, but by varying degrees. The first definition of importance is linear and equal to the product of ROS and time since last visit:

$$Importance = ROS * (t_{current} - t_{last\ visit})$$

where ROS is rate of spread, $t_{current}$ is the current simulation time, and $t_{last\ visit}$ is the time at which the cell was last visited. This definition of importance will be referred to as "linear importance".

The second definition of importance is exponential. It is defined as follows:

$$Importance = (t_{current} - t_{last\ visit})^{1+ROS}$$

The variables have the same definition as in the linear case. The addition of 1 to ROS in the exponent amplifies the effect of ROS.

The definition of importance should reflect the simulation designer's goal in data assimilation and does not necessarily need to be one of the two definitions above. These two definitions were chosen because they vary at two different rates for the same ROS. The linear importance varies linearly with the time since last visit and returns importance values directly proportional to ROS, so two cells with very different ROS values will not differ greatly in importance. The exponential definition of importance returns importance values with higher difference for the same two cells. It also varies exponentially with time, so a cell with a higher ROS becomes more important faster than a cell with a low ROS, as seen in Figure 3.2. The results of experiments using both definitions of importance are detailed in Section 5.

One of the intended consequences of both definitions is that cells with higher ROS values will be visited more frequently than cells with low ROS values because those cells with high ROS values will grow in importance faster than low ROS cells. The other important consequence is that every cell will have an importance of zero at the exact moment a UAV visits the cell because $t_{current}=t_{last\ visit}$. The UAV is lowering the total importance of its region by visiting the cells, and this has implication on the path planning and negotiation algorithms. The designer of the data assimilation model should define the importance according to how much value is derived from data in locations with high inherent uncertainty (ROS) and how quickly that value grows with time.

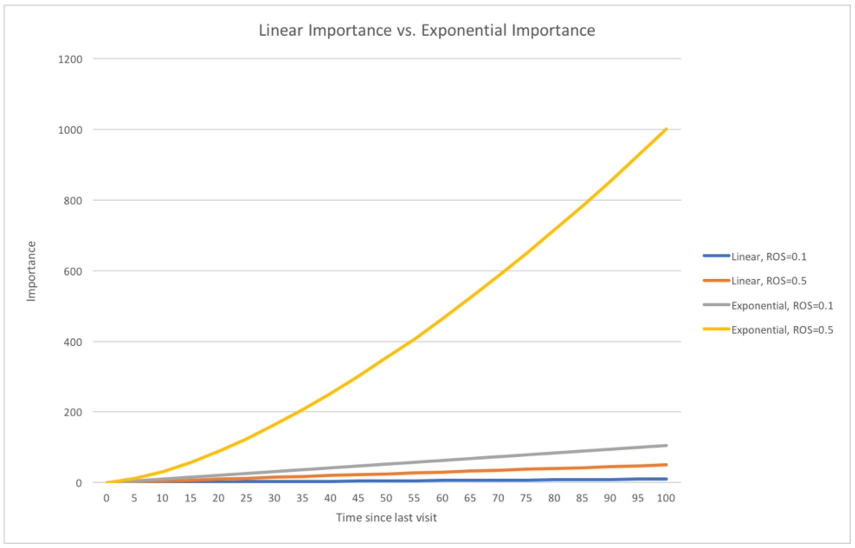


*Figure 3.2 Comparison of Linear Importance with Exponential Importance*

## 3.3 Cost Function

The effectiveness or efficiency of the UAV team's strategy is determined by how frequently the UAVs visit cells of varying importance. The quality of the data collected for assimilation is directly proportional to the importance of the cell from which the data is collected. Cells with higher importance should be visited first. The importance of a cell can also be thought of as the cost of not visiting that cell. As the importance of the cell grows with time, the cost of not visiting that cell also grows. So the goal of the UAV team is to maintain the lowest possible average cost – or average importance - over all the cells in the wildfire perimeter. Because the importance drops to zero when a UAV visits that cell, visiting the highest importance cell lowers the average importance more than visiting any other cell. This is the guiding principle in the design of the path planning and negotiation algorithms in the following section. The following cost function is defined as the average importance value for the whole perimeter.

$$Cost = \frac{\sum_{i=1}^{i=n} imp_i}{n}$$

where *n* is the number of cells in the perimeter, *i* is the index of one cell in the perimeter, and $imp_i$ is the importance value for cell *i*.

## 4 SOLUTION AND ALGORITHMS

In the previous section, the cost function was defined as the average importance of the cells in the perimeter. The primary objective of the UAV team and each individual UAV is to maintain the lowest cost possible at each time step. The secondary objective is to maintain this low cost with the least amount of variation. A cost function that varies linearly is more desirable than one that fluctuates wildly over time.

Each UAV in the team is responsible for visiting a one particular region of cells. The region is a contiguous segment of "burning" cells along the perimeter of the wildfire. As previously discussed, each cell has an importance value proportional to its ROS and the time since the UAV last visited the cell. The objective of the UAV team is to minimize the cost function over the entire perimeter, so the objective of the individual UAV can be construed as minimizing the cost function for its own region. Indeed, if all UAVs minimize the cost of their regions, then the cost of the perimeter, which is the average of the regions' costs, should be minimized as well. It turns out that the team objective can be better accomplished by more than just individual action; cooperation between UAVs in the form of negotiation can create divisions of regions that are better suited for the individual efforts, which will be explained in subsection 4.2. However, once the regions are divided most effectively, the best individual UAV strategy should be the one that minimizes the cost function for the region. The path planning algorithms designed to achieve the minimum cost function are detailed in the following subsection.

## 4.1 Path Planning Algorithms

The individual UAV’s region can be imagined as a linked list of cells with varying importance, and the list of cells may change in length and value dynamically. The UAV is only allowed to move one cell in either direction at a time, and its location is constrained to the cells that make up the list. The UAV must choose one-dimensional route that has two parameters: direction (clockwise or counter-clockwise) and number of cells to traverse in that direction. The method for setting these parameters at a given time can affect the value of the region’s cost function at a future point in time. The following path planning algorithms were devised to minimize the cost function in a single UAV’s region.

### *4.1.1 Back and forth*

The first path planning algorithm is the simplest. It is called the back and forth algorithm, and follows these steps:

```
cw = true;
for each cycle do
    if cw and uav at last cell in region
        cw = false;
    else if cw=false and uav at first cell in region
        cw = true;
    set last visit time of current cell to the current time;
    if cw
        move UAV forward by one cell;
    else if ccw
        move UAV back by one cell;
end for
```

The meaning of "each cycle" is at the next movement event scheduled for the UAV, when the UAV is considered to be at the next cell. The *cw* variable controls the direction of the UAV. If *cw* is true, then the UAV moves clockwise, or towards the end of the list of cells in the region; if *cw* is false, the UAV move counter-clockwise back towards the beginning of the list. The UAV traverses the cells in its region sequentially until it reaches the very first or last cell in the region, depending on which direction it is moving, at which point the UAV changes direction and repeats the process.

This algorithm is naïve, but it is effective because the UAV following these steps will visit every cell in its region once during its traversal. The cell in the middle of the region is visited every *t* while the cells at the extreme ends of the region are visited every *2t*, where *t* is the time it takes the UAV to traverse the region once. If all the cells in the region have the same ROS, then the back and forth algorithm performs just as well as any other path planning algorithm. However, if there is an uneven distribution of ROS values in the region, some parts of the region will grow in importance faster than the parts with lower ROS, and the back and forth algorithm will produce less than ideal results, especially if the cells near the ends of the region have the highest ROS.

### 4.1.2 Highest Importance Target Dynamic

The first attempt at improving the back and forth algorithm involves finding the cell with the highest importance in the region every time the UAV moves to a new cell and setting that cell as the target.

```
currentCell = cell where UAV is currently located;
for each cycle do
    set last visit time of currentCell to the current time;
    targetCell = cell with highest importance in region;
    cw = index of targetCell > index of currentCell;
    if cw
        currentCell = cell at index of currentCell + 1;
    else if ccw
        currentCell = cell at index of currentCell – 1;
end for
```

Each time the UAV reaches a new cell, the last visit time of the UAV's current cell is set to the current time, and the highest importance cell in the region is set as the target cell. The UAV's direction is set toward that target cell, and the it moves forward or backwards by one cell depending on which direction the target cell is located.

The intention of this algorithm is that the UAV will always be moving towards the highest importance cell in an effort to minimize the cost function by visiting the highest importance cell as quickly as possible. Despite the intention, this algorithm may perform worse than the back and forth algorithm because it neglects the importance of the cells in between the UAV's current cell and the target cell. It also updates the target cell each time the UAV moves to a new cell, regardless of whether the UAV has reached its initial target or not. This condition can potentially cause the UAV to get stuck in a loop of traversing a small subregion of cells if the two highest importance cells lie on either side of that subregion and grow in importance at varying rates. The UAV will keep changing direction when the cell in the opposite direction becomes the highest importance cell, traversing back over cells with low importance because it has just visited them.

### *4.1.3 Highest Importance Target Static*

This algorithm is similar to the previous highest importance target dynamic algorithm, but instead of setting the target as the highest importance cell every time the UAV moves to a new cell, the target is only updated once the UAV has reached the original target.

```
currentCell = cell where UAV is currently located;
targetCell = cell with highest importance in region;
for each cycle do
    set last visit time of currentCell to the current time;
    if currentCell == targetCell
        targetCell = cell with highest importance in region;
        cw = index of targetCell > index of currentCell;
    if cw
        currentCell = cell at index of currentCell + 1;
    else if ccw
        currentCell = cell at index of currentCell – 1;
end for
```

This variation of the highest importance target algorithm relies on the assumption that the highest importance cells will be clustered together, and on its way to the highest importance target cell the UAV will visit other high importance cells. This scenario is more likely if the time since last visit plays a more important role in determining importance than does ROS. After the UAV reaches the initial target cell, it searches the region for the new highest importance cell, which is possibly beyond the old target cell continuing in the same direction, or it could be backwards beyond the cells the UAV has just traversed. The current cell should never be returned as the highest importance cell upon search for a new target cell because its importance value is always zero, and the other cells in the region will have non-zero importance as long as their ROS is non-zero.

### *4.1.4 Predictive Planning*

The final path planning algorithm is the most computationally expensive but is expected to provide the most effective results. The predictive planning algorithm searches through all possible paths that do not involve any change of direction, other than the initial move, and chooses the path with the highest cost function. The process will be elaborated shortly, but the strength in this algorithm is in its predicting the importance of each cell at a future time when the UAV would visit that cell, and adds that value to the cost function – instead of the current importance of the cell. For example, if the path being predicted contained three cells, the cost function would be:

$$Cost = \frac{ROS_1(t_{current} - t_{last\ visit}) + ROS_2((t_{current} + 1) - t_{last\ visit}) + ROS_3((t_{current} + 2) - t_{last\ visit})}{3}$$

where $ROS_1$ is the ROS of the first cell in the sequence (the cell that the UAV would visit first), $ROS_2$ is the ROS of the cell that the UAV would visit next, and so on. The addition of incremented integers to $t_{current}$ at each sequential cell is a simplification of the actual time it would take the UAV to travel between cells, but it illustrates the point. The following pseudocode outlines the procedure:

```
currentCell = cell where UAV is currently located;
for each cycle do
    set last visit time of currentCell to the current time;
    highestValue = cost function of path from currentCell to targetCell;
    for each cell ‘destination’ in region do
        cost = predicted cost function of path from current cell to destination;
        if cost > highestValue
            targetCell = destination;
            cw = index of targetCell > index of currentCell;
            highestValue = cost;
    end for
    if cw
        currentCell = cell at index of currentCell + 1;
    else if ccw
```

```
        currentCell = cell at index of currentCell – 1;
end for
```

This predictive algorithm is another variation of the original highest importance target dynamic algorithm in that it calculates the best path every time the UAV reaches a new cell. It avoids the problem of continuously traversing back and forth over low importance cells by considering every possible path instead of only the direction of the highest importance cell. This rules out the traversal of recently visited cells as long as the cells ahead of the UAV have a higher average cost than the cells the UAV has just visited. The prediction of cells' importance values at the time the UAV would visit them allows the UAV to consider the weight of cells with high *inherent* importance (high ROS) even if they were very recently visited and have a low current importance.

## 4.2 Negotiation Algorithms

In addition to individual UAV behavior, the cooperative behavior of the team can provide tremendous gains in efficiency. The perimeter of the fire is initially divided into regions of equal size and assigned to each UAV. Each UAV starts out responsible for the same number of cells, but the number of cells is not a good measure of responsibility when the cells have varying importance values. Two UAVs with regions of equal size may visit all of their cells with similar frequency, but the cells in the two regions may differ in ROS and require very different frequencies of observation. So, after the perimeter is divided into equal lengths among them, the UAVs must follow a negotiation procedure to more evenly spread the responsibility based on the cost function.

### *4.2.1 Negotiate on Importance*

The most obvious method for negotiating the transfer of cells between UAVs is to redistribute the importance values so that they are most evenly spread between all UAVs. This may result in some UAVs being responsible for a larger section of the wildfire than others, but the cost functions for each UAV's region will be as similar as possible. The following pseudocode outlines the negotiation procedure based on importance:

```
at each negotiation interval do
    for each active UAV 'uav' do
        for each other active UAV 'other_uav' do
            if other_uav is a neighbor and other_uav's region total importance
                 < uav's region total importance
                add other_uav to key of map 'neighbors';
                add total importance of other_uav to value of map 'neighbors';
        end for
        determine offers based on size of 'neighbors';
        for each 'neighbor' in map 'neighbors' do
            chunk = empty list of cells;
            cellToTransfer = uav's cell that borders neighbor's region;
            while total importance of chunk <= offer to neighbor AND cellToTransfer
                does not contain uav, uav's next cell, or uav's target cell
                add cellToTransfer to chunk;
                if total importance of chunk > offer to neighbor
                    remove cellToTransfer from chunk;
                    break out of while loop;
                cellToTransfer = uav's cell neighboring current cellToTransfer;
            add chunk to neighbor's region;
            remove chunk from uav's region;
        end for
    end for
```

where the *negotiation interval* can be defined as any time interval. The shorter the interval, the more often the UAVs will negotiate and the more stable the distribution of cost will remain. The longer the negotiation interval, the more imbalanced the levels of responsibility between UAVs can become. After the negotiation interval elapses, each UAV checks the total importance value of its neighbors and compares them with its own. If that UAV is currently responsible for a higher total importance than the neighbor, then that neighbor is added to the key of a map of neighbors (up to two), and the total importance of the neighbor's region is added as the value for that key.

Based on the number of neighbors with lower importance levels, the UAV determines offers for transfer to its neighbor(s). If the UAV has two neighbors with lower importance, it uses the following algorithm to determine the two offers:

```
neighbor1Importance = importance of the first 'neighbor' in neighbors
neighbor2Importance = importance of the second 'neighbor' in neighbors
delt1Imp = thisUAVimportance - neighbor1Importance;
delt2Imp = thisUAVimportance - neighbor2Importance;
scale; (for scaling importance difference to appropriate offer)
if (delt1Imp < delt2Imp)
{
    scale = delt1Imp / (2*delt1Imp + delt2Imp);
} else {
    scale = delt2Imp / (2*delt2Imp + delt1Imp);
}
offer1 = scale * delt1Imp;
offer2 = scale * delt2Imp;
```

where *thisUAVimportance* is the total importance of the UAV calculating the offers. The *scale* variable is determined by the following equation:

$$scale = \frac{imp_{uav} - imp_{high\ neighbor}}{2 * (imp_{uav} - imp_{high\ neighbor}) + \ (imp_{uav} - imp_{low\ neighbor})}$$

where $imp_{high\ neighbor}$ is the total importance value of the UAV's neighbor with the highest total importance, and $imp_{low\ neighbor}$ is the total importance value of the UAV's neighbor with the lowest total importance. When the differences between the UAV's importance and its neighbors' importance values are multiplied by *scale*, the two offers will bring the UAV's importance down so that it is equal to the importance of the higher importance neighbor after the transfer, and the offers will be weighted by the respective differences in importance.

If the UAV has only one neighbor that is responsible for less importance than the UAV, then the UAV offers an amount that will make the two UAVs' importance values equal after transfer. If the UV has no neighbors with less importance, then it simply makes no offers.

After the offers are determined, the UAV making the offers begins to add bordering cells to a chunk of cells that are to be transferred to the neighbor. The UAV continues adding cells to the chunk sequentially moving away from the border until the total importance of the chunk exceeds the value of the offer or the cell being added to the chunk contains either the UAV, the UAV's next cell, or the UAV's target cell. If the last cell added to the chunk brings its total importance over the value of the offer, the cell is removed from the chunk. Finally, the chunk is removed from the UAV's region and added to the neighbor's region.

This negotiation procedure attempts to redistribute the importance levels evenly amongst all UAVs; however, one iteration through this negotiation procedure is not guaranteed to completely even out the importance levels of all the regions. When one UAV ends up with more importance than all the other UAVs, it transfers as much as it can to its two neighbors without transferring so much that itself would become responsible for less than the neighbors. It then takes additional iterations of the negotiation procedure for that extra importance to propagate through the team to the other UAVs that do not neighbor the original UAV, but, with each iteration, any excess importance value eventually evens out like the ripples in a body of water. The algorithm may never fully converge; it may keep transferring a small remainder of importance until the next big imbalance in importance occurs. In the extremely dynamic wildfire simulation, convergence of the negotiation algorithm is of little concern because there are constant fluctuations in the importance values.

### *4.2.2 Negotiate on Center of Importance*

The negotiate on importance algorithm performs very well given its intention, but it is based solely on the *total* importance of the UAV regions and fails to consider the distribution of the individual cells' importance values within the region. This algorithm was developed after testing all of the path planning algorithms, which all turn out to cause behavior nearly identical to the basic back and forth algorithm in most wildfires. As mentioned previously, the cell at the center of a region is visited every *t* time units, while the two cells at either end of the region are visited once every *2t* time steps, given the UAV is following the basic back and forth path planning algorithm.

This means that even with a path planning algorithm at work to keep the UAV visiting most important cells more frequently, the nature of the one-dimensional path most often causes the UAV to traverse the entire region back and forth completely, causing the cells nearest to the center of the region to visited most frequently. The cells further from the center are still visited with the same average frequency as those at the center (except for the two at the very ends), but the standard deviation becomes larger the further the cell is from the center of the region.

Since the path planning algorithms have little effect on the frequency of visitation for higher importance cells, the negotiation procedure is modified to skew the range of each region so that the highest importance cells are located nearest the middle point of the region. The concept of center of gravity is adapted to calculate the "center of importance" of a region.

$$Center\ of\ Importance = \frac{\sum_{i=1}^{i=n} imp_i * i}{totalImp}$$

where *$imp_i$* is the importance of cell *i* in the region (*i* is the index of the cell within the region list) and *totalImp* is the total importance of the region. The negotiation on center of importance attempts to transfer cells not only to redistribute importance amongst the UAVs, but also to align each region's center of importance as closely as possible with its center cell.

This modified negotiation follows the same procedure as the negotiation on importance until its begins adding cells to the chunk to be transferred; the offers are calculated using the same scaling method. Once the negotiation algorithm determines the offers, depending on the number of neighbors with lower importance levels, it calls a "swap on center of importance" algorithm. The algorithm searches through every combination of offers (within the bounds of the original offers determined by total importance) and returns the offers that minimize the difference between the resulting region's center of importance and the index of its center cell. The pseudocode for the algorithm follows:

```
chunk1 = maximum chunk allowed by offer to neighbor1
chunk2 = maximum chunk allowed by offer to neighbor2
bestCoi = center of importance – index of center cell;
optimalChunk1 = empty list;
optimalChunk2 = empty list;
if chunk1 is not empty
        while chunk1 is not empty do
                if chunk2 is not empty
                        while chunk2 is not empty do
                                cellsToRemove = empty list;
                                add all cells in chunk2 to cellToRemove;
                                if center of importance of (region minus cellsToRemove) – center
                                        index of (region minus cellsToRemove) < bestCoi
                                                add all cells in chunk1 to optimalChunk1;
                                                add all cells in chunk2 to optimalChunk2;
                        bestCoi = center of importance of (region minus
                            cellsToRemove) – center index of (region
                            minus cellsToRemove);
                    remove last element (cell) from chunk2;
                remove last element from chunk1;
            else if center of importance of (region minus chunk1) – center index of
                (region minus chunk1) < bestCoi
                add all cells in chunk1 to optimalChunk1;
                add all cells in chunk2 to optimalChunk2;
                bestCoi = center of importance of (region minus chunk1) – center index of
                    (region minus chunk1);
            remove last element from chunk1;
    else if chunk2 is not empty
        while chunk2 is not empty do
            else if center of importance of (region minus chunk2) – center index of (region
                minus chunk2) < bestCoi
                add all cells in chunk1 to optimalChunk1;
                add all cells in chunk2 to optimalChunk2;
                bestCoi = center of importance of (region minus chunk2) – center index of
                    (region minus chunk2);
            chunk2.remove(chunk2.size() - 1);
    for each cell 'cellToRemove' in optimalChunk1 do
        remove cellToRemove from region;
    end for
    for each cell 'cellToRemove' in optimalChunk2 do
        remove cellToRemove from region;
    end for
    add optimalChunk1 to neighbor1 region;
    add optimalChunk2 to neighbor2 region;
```

This algorithm is for the case when the UAV has two neighboring UAVs, *neighbor1* and *neighbor2*, both with less total importance than the UAV. If the UAV only has one neighbor with less total importance, a similar procedure is followed to determine the one *optimalChunk* that will result in the region with the center of importance nearest to the center cell. If the UAV has no neighbors with lower importance, it will not transfer any cells.

## 4.3 Benchmark Algorithms

Two benchmark algorithms are implemented and tested for comparison with the efficiency of each combination of path planning and negotiation algorithms. The benchmark cases do not use any negotiation or advanced path planning algorithms. The first benchmark algorithm, the "circling" algorithm, involves the team of UAVs all travelling clockwise around the perimeter of the wildfire. The UAVs begin at points equidistant from each other, but because the wildfire dynamically grows in some sections faster than others, the distance between the UAVs will vary over time.

The second benchmark algorithm, the "bumper" algorithm, deploys each UAV in the team at equidistant points along the perimeter with each UAV travelling in a direction opposite to its neighbor on the counter-clockwise side. After the UAVs begin moving, they will eventually run into their neighbors, at which point they change direction and continue until the other neighbor is encountered, change direction again, and repeat the process. Neither of the benchmarks consider frequency of visit or division of responsibility, and therefore provide a good base against which to compare the newly developed algorithms.

# 5 EXPERIMENTS AND RESULTS

## 5.1 Experiments

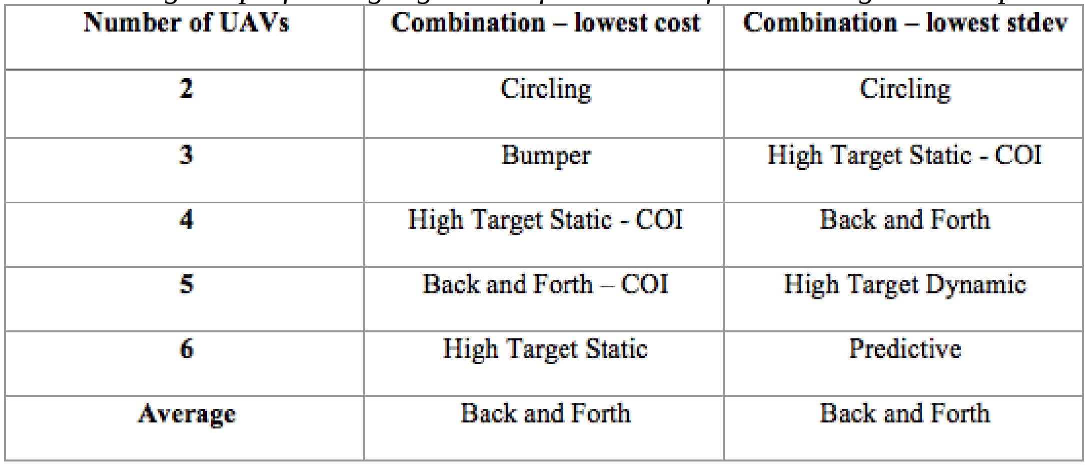
The path planning algorithms and negotiation algorithms described in the previous section are implemented as part of a UAV simulation model in DEVS-FIRE environment. Every combination of path planning algorithm, negotiation algorithm, and importance definition (linear or exponential) is tested with a varying number of UAVs (2-6) for a total of 80 simulations. The two benchmark algorithms are implemented using both exponential and linear importance and tested with a varying number of UAVs (2-6) for a total of 20 benchmark simulations. The UAVs travel with the same speed in all simulations and are modeled to be able to change direction instantly. The global cost function for the entire wildfire perimeter as well as the cost functions for the individual UAVs' regions are collected every 20 time steps (20 seconds of simulation time).

The purpose of simulating every possible combination of path planning and negotiation algorithm is to determine which combination minimizes the global cost function and which combination maintains a stable cost function. Each combination of algorithm is tested using both the linear and exponential definition of importance. Changing the definition of importance gives an idea of the behavior of the algorithms under different objectives for data assimilation. For each combination, the average and the standard deviation of the cost function are taken over time. The lower the average, the more efficient the data collection strategy. Likewise, the lower the standard deviation, the more stable the cost function, meaning the data collection strategy is collecting data from cells with appropriate frequency.

## 5.2 Linear Importance

A total of 50 simulations, including 10 benchmark cases, were performed using linear importance. The average cost and standard deviation for each combination of algorithms and benchmarks are charted and compared. The results for each combination are shown in Table 5.1. Surprisingly, the benchmark cases performed better with a small number of UAVs; the advanced algorithms began showing better performance when 4 or more UAVs were used to monitor the wildfire. The likely reason for this is that the path planning and negotiation algorithms caused the UAVs to visit recently visited cells more often than the benchmark cases. The average cost and standard deviations were averaged over every number of UAVs, showing the Back and Forth algorithm using Negotiate on Importance performed best on average. The average values are shown in Figure 5.1. There is little difference in the performance of each combination using the linear definition of importance. This is to be expected as the linear definition assigns similar importance values to cells with varying ROS.

*Table 5.1 Highest performing algorithms for number of UAVs using linear importance*



| **Number of UAVs** | **Combination – lowest cost** | **Combination – lowest stdev** |
|---|---|---|
| **2** | Circling | Circling |
| **3** | Bumper | High Target Static - COI |
| **4** | High Target Static - COI | Back and Forth |
| **5** | Back and Forth – COI | High Target Dynamic |
| **6** | High Target Static | Predictive |
| **Average** | Back and Forth | Back and Forth |

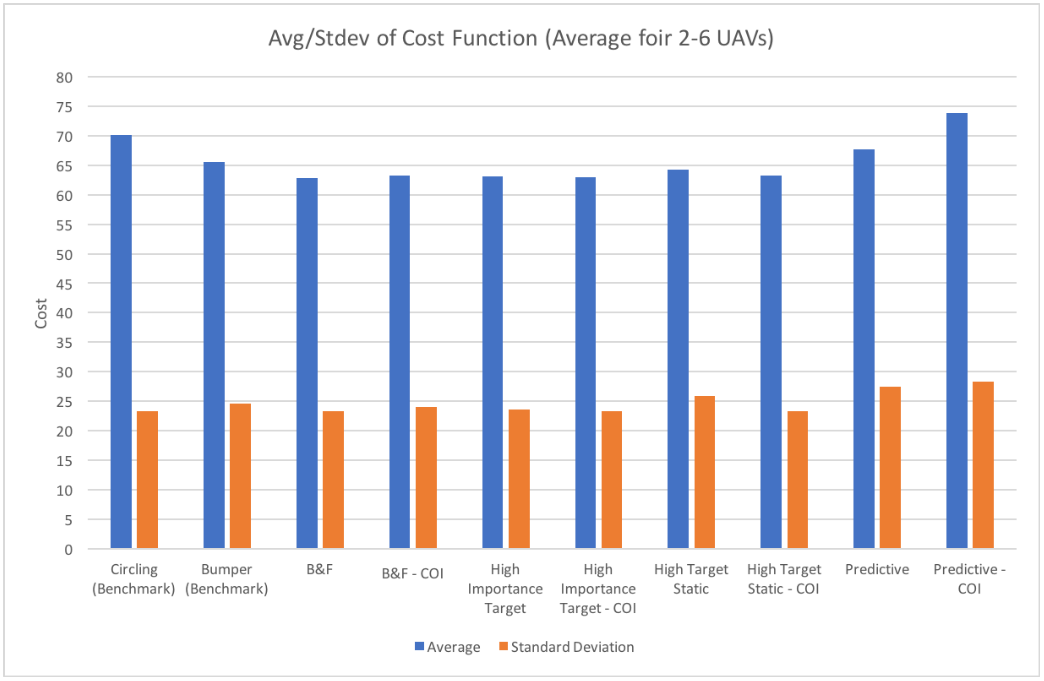


*Figure 5.1 Efficiency of algorithm combinations using linear importance*

## 5.3 Exponential Importance

Another 50 simulations were performed using the exponential definition of importance. The average cost and standard deviation for each combination of algorithms and benchmarks are charted and compared. The results for each combination are shown in Table 5.2. Again, the benchmark cases performed better with a small number of UAVs, while the advanced algorithms showed better performance with 4 or more UAVs. The average cost and standard deviations were averaged over every number of UAVs, shown in Figure 5.2. The High Importance Target Dynamic path planning algorithm had the best performance, and did so with the lowest standard deviation using the Negotiate on Center of Importance algorithm. The higher variance in performance is expected from using exponential importance because the disparity in the growth rates of importance for cells with varying ROS values is much greater than when using the linear definition. With the lower disparity from using linear importance, all the path planning algorithms cause nearly the same behavior as the back and forth algorithm. This is not the case when using exponential importance; the higher disparity brings out the intended behavior of the path planning algorithms.

*Table 5.2 Highest performing algorithms for number of UAVs using linear importance*

| Number of UAVs | Combination – lowest cost | Combination – lowest stdev |
|---|---|---|
| 2 | Circling | Circling |
| 3 | Bumper | Bumper |
| 4 | High Target Dynamic | High Target Dynamic - COI |
| 5 | Predictive | High Target Static - COI |
| 6 | High Target Static | High Target Static |
| Average | High Target Dynamic | High Target Dynamic - COI |

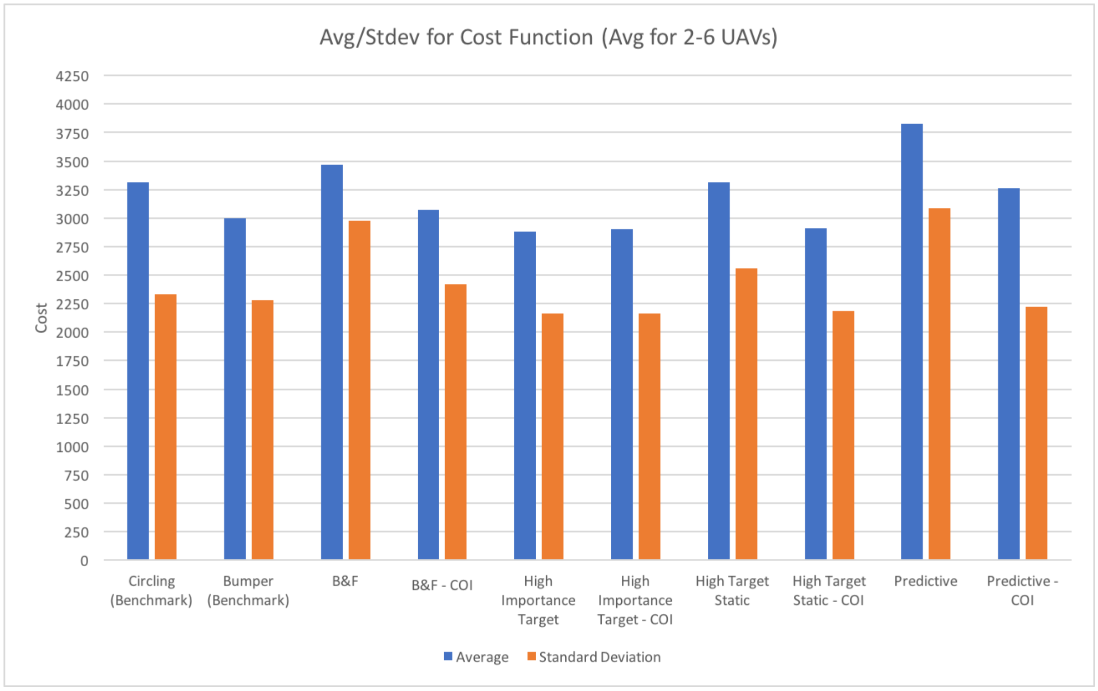
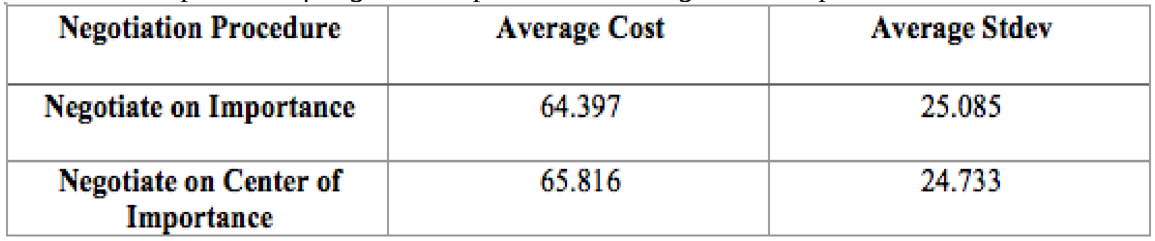


*Figure 5.2 Efficiency of algorithm combinations using exponential importance*

## 5.4 Comparison Negotiation Procedures

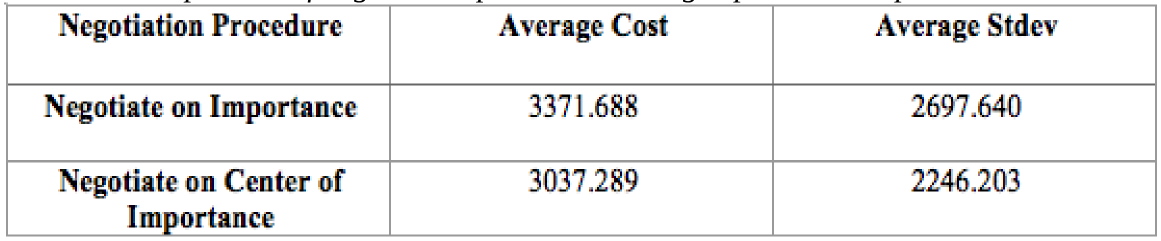
While the path planning algorithms achieved somewhat marginal improvements over the benchmarks, the addition of multiple UAVs using procedure negotiation clearly improved performance. Of the two negotiation procedures, the Negotiate on Center of Importance was expected to yield the lowest cost as wells as the lowest standard deviation. The results shown in Tables 5.1 and 5.2 contradict that expectation; however, looking at the average cost and standard deviation for every number of UAVs using the two negotiation procedures, it is clear the Negotiate on Center of Importance does outperform Negotiate on Importance on average. Table 5.3 shows the comparison between the two negotiation procedures using linear importance, and Table 5.4 shows the comparison using exponential importance. With both definitions of importance, the Negotiate on Center of Importance algorithm produced lower average standard deviations. The Negotiate on COI algorithm reduced the average cost more when using exponential importance, while Negotiate on Importance reduced the average cost slightly more when using linear importance.

*Table 5.3Comparison of negotiation procedures using linear importance*



| Negotiation Procedure | Average Cost | Average Stdev |
|---|---|---|
| Negotiate on Importance | 64.397 | 25.085 |
| Negotiate on Center of Importance | 65.816 | 24.733 |

*Table 5.4Comparison of negotiation procedures using exponential importance*



| Negotiation Procedure | Average Cost | Average Stdev |
|---|---|---|
| Negotiate on Importance | 3371.688 | 2697.640 |
| Negotiate on Center of Importance | 3037.289 | 2246.203 |

## 5.5 Number of UAVs

Each combination of path planning, negotiation algorithm, and importance definition is simulated for teams ranging from 1 to 6 UAVs. The results of the using the Back and Forth path planning algorithm in combination with the Negotiate on Importance algorithm are shown for each number of UAVs in Figure 5.3 (the sharp changes in cost are due to changes in the weather model). As expected, using additional UAVs to collect data reduces the cost function and its standard deviation. However, the important consequence of these results is that the amount the cost function is reduced with the addition of another UAV decreases as the number of UAVs increases. There is a tradeoff between greater efficiency of data collection and the cost of adding more UAVs to the team. Therefore, the user of this data assimilation method must make an optimization decision regarding the number of UAVs that can be purchased and operated within a budget against the added benefit of that number of UAVs.

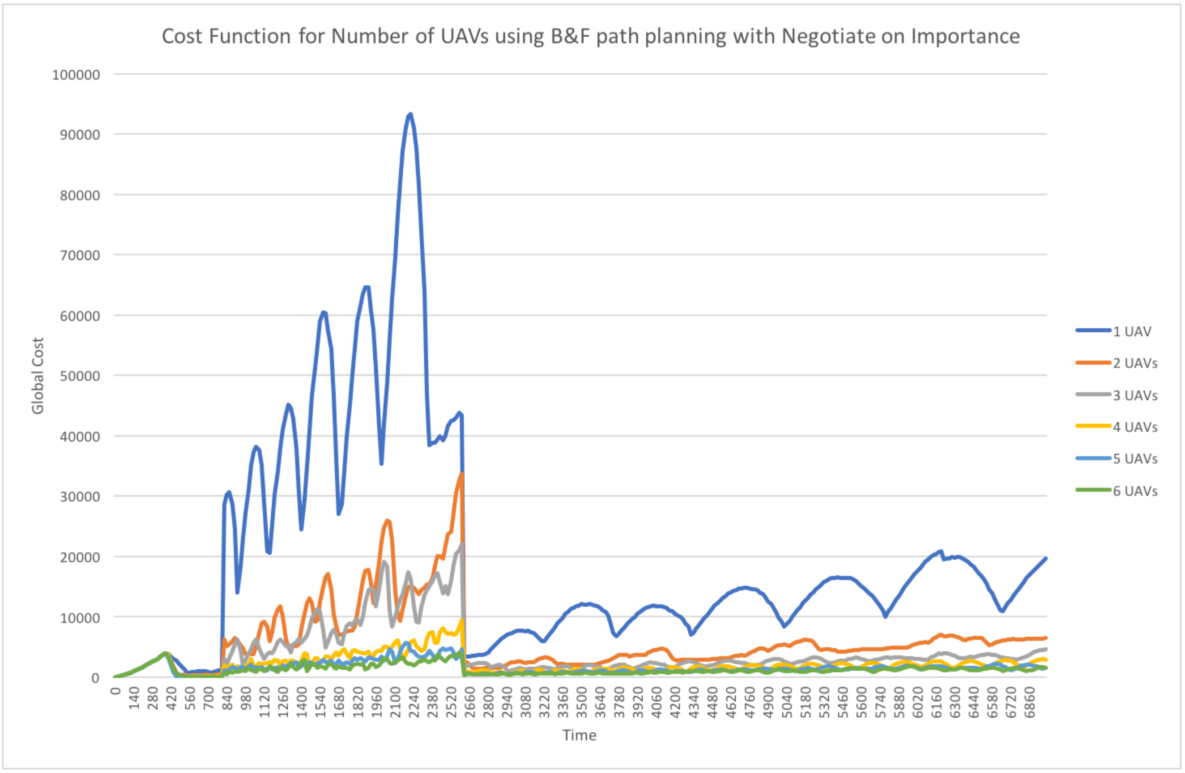


*Figure 5.3 Efficiency of multiple-UAV teams*

# 6 CONCLUSIONS

The efficiency of data assimilation for dynamic data driven simulation of wildfires can be improved by utilizing a team of UAVs that are controlled by the path planning algorithms and negotiation algorithms developed in this thesis. The varying levels of importance of the data being collected is a central concept to the methodology of data assimilation, and the mathematical definition of importance chosen must reflect the goals of the simulation user. The algorithms developed herein are applicable to any definition of importance but respond differently to each one, and this is the most valuable feature of the data assimilation methodology. This thesis serves as a basis for data assimilation into any simulated system using a team of autonomous mobile robots. The path planning and negotiation algorithms can be easily adapted to any scenario in which the data of interest lies on a dynamic or static perimeter.

## 6.1 Future Work

There are some possible improvements that can be made to both the path planning and the negotiation algorithms. For instance, the predictive path planning algorithm recalculates the path with the highest cost function every time the UAV moves to a new cell. This algorithm can be modified to only recalculate the highest-valued path once the UAV completes the last computed path entirely. Another negotiation algorithm could be designed to consider the amount of time before each UAV would be able to visit a cell as the deciding factor on which UAV would be responsible. This algorithm is more closely related to the negotiation algorithm used in [8].

The path planning and negotiation algorithms should next be implemented using a UAV autopilot software, such as Paparazzi UAV, which provides a Hardware-in-the-Loop (HITL) simulation environment for simulating the algorithms with real UAV hardware. Once the UAV team is successfully simulated using HITL approach, the UAVs can be deployed into a real system for testing.

Beyond the wildfire system, the data assimilation methodology described in this thesis can be further formalized in a more general scope. Wider and finer definitions of importance should be tested and characterized. This will allow a data assimilation framework to be developed, giving dynamic data driven simulation a greater foothold in practice.

## REFERENCES

[1]Westerling, A.L. 2016. Increasing western U.S. forest wildfire activity: Sensitivity to changes in the timing of spring. Phil. Trans. R. Soc. B. 371:20150178.

[2]NIFC (National Interagency Fire Center). 2016. Total wildland fires and acres (1960–2015). Accessed March 2016. www.nifc.gov/fireInfo/fireInfo_stats_totalFires.html.

[3]Xiaolin Hu and Yi Sun, "Agent-based modeling and simulation of wildland fire suppression," *2007 Winter Simulation Conference*, Washington, DC, 2007, pp. 1275-1283.

[4]Meroney, Robert N. "Fire whirls and building aerodynamics." *Proceedings of the 11th International Conference on Wind Engineering*. 2003.

[5]Hu, Xiaolin. "Dynamic data driven simulation." *SCS M&S Magazine* 5 (2011): 16-22.

[6] Advantages of UAS https://www.uavs.org/advantages

[7] Ahmadi, Mazda, and Peter Stone. "Continuous area sweeping: A task definition and initial approach." In *Advanced Robotics, 2005. ICAR'05. Proceedings., 12th International Conference on*, pp. 316-323. IEEE, 2005.

[8] Ahmadi, Mazda, and Peter Stone. "A multi-robot system for continuous area sweeping tasks." In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 1724-1729. IEEE, 2006.

[9] Ntaimo, Lewis, Xiaolin Hu, and Yi Sun. "DEVS-FIRE: Towards an integrated simulation environment for surface wildfire spread and containment." *Simulation* 84.4 (2008): 137-155.

[10] Hu, Xiaolin, Yi Sun, and Lewis Ntaimo. "DEVS-FIRE: design and application of formal discrete event wildfire spread and suppression models." *Simulation* 88.3 (2012): 259-279.