Optimizing Border Patrol Operations Using Unmanned Aerial Vehicles

Doina Bein, Wolfgang Bein, Ashish Karki
Department of Computer Science
University of Nevada, Las Vegas
Las Vegas, USA
wolfgang.bein@unlv.edu

Bharat B. Madan
Dept. of Modeling, Simulation, and Visualization Eng.
Old Dominion University
Norfolk, USA
bmadan@odu.edu

***Abstract*—The most common approach for border patrol operations is the use of human personnel and manned ground vehicles, which is expensive, at times inefficient and sometimes even hazardous to people involved. A better approach would be using Unmanned Aerial Vehicles (UAVs) in combination with such ground sensors. This would help improve the overall effectiveness of the surveillance system as a UAV could first scan the alert area before sending in personnel and vehicles, if deemed necessary. We propose border surveillance using multiple Unmanned Aerial Vehicles (UAVs) in combination with alert stations consisting of Unattended Ground Sensors (UGSs) along the border line/fence. Upon detecting an event, an alert would be triggered by any UGS. We simulate this process by reading probability data for different timestamps from a text file. And, based on utility values of each station, two UAVs decide on which alert station to service.**

Keywords-border surveillance, unmanned aerial vehicle, optimization, probability of detection.

I. INTRODUCTION

The border of a country defines its political boundaries and areas of its jurisdictions. A border region carries immense importance for a country in terms its national security. But, proper controlling of such regions is a major challenge for most of them. These boundaries not only contain land areas but also water bodies and rough territories. Also, the length of the boundary line may sometimes be long and patrolling it puts a strain on the resources that any nation can afford. As an example the total length of border between the U.S. and Mexico is approximately 2000 miles and that between the U.S. and Canada is approximately 5500 miles. Sometimes security may be improved by fencing a part of the border line. But, fencing alone is not enough. These fences need to be patrolled. Ground sensors can sometimes be a part of such fencing or could be used separately. Such sensors trigger alarms at possible intrusion. False alarms require unnecessary human intervention which in turn is expensive and even dangerous.

We assume the border to be a terrestrial area long and thin (relative to actual border size). There are a number of alert stations consisting of Unattended Ground Sensors (UGSs). Any UGS, upon detecting an intrusion of any type, sets off an alert. A UAV is then sent to this particular site or station to further investigate the alert.

The paper is organized as follows. In Section 2 we present related work, relevant to our border monitoring problem. In Section 3 we present a proposed algorithm by which two UAVs monitor a number of alert sites with the help of UGSs. In Section 4 we present the software program used for obtaining simulation results. In our simulations, values expressed in terms of percentage for different types of events (detected by a UGS) are extracted from a text file. The percentages represent the probability of occurrence of each type of event for a particular site. In addition, there is intelligence data file containing coefficient values for the same types of events. We conclude in Section 5.

II. RELATED WORK

Our method proposed for optimizing the use of UGSs and UAVs for border patrolling could be extended to Perimeter Patrol Operations as well. The application of such operations could be securing a nuclear facility or a military installation and monitoring a wildlife reserve or an oil field [2].

A perimeter patrol problem consisting of multiple UAVs equipped with cameras and controlled remotely by an operator is addressed in [3]. The decision problem solved is the determination of optimal loitering or dwelling time of a UAV at an alert site. That is the optimal amount of time spent at a site so that maximum amount of information can be gathered but at the same time the servicing delay to other sites is kept as minimal as possible. The problem is formulated as a Markov decision process and a solution is obtained using dynamic programming. Simulations are run for cases like using one or two UAVs and UAVs with or without turnaround capabilities.

A system to coordinate the working of multiple UAVs to be utilized in border and perimeter patrolling jobs is presented in [4]. The motion control and navigation among the UAVs is divided into a hierarchical architecture. The lower hierarchy consists of controllers that perform positioning and tracking jobs. The higher hierarchy systems are used for maneuvering operations. A team of UAVs is assigned to patrol a particular region of a border. A border region is defined to be a geographical region which is, in general, like a line: thin and long. A particular border region is split into sectors which are roughly the same in shape and size. Each aircraft is assigned a sector. If any UAV detects an event or target, it alerts its operator. If the operator deems the target is important, the UAV

978-1-4799-8828-0/15 $31.00 © 2015 IEEE
DOI 10.1109/ITNG.2015.83

is commanded to follow it. The sectors are then dynamically reassigned among remaining UAVs.

Currently, UAVs also called drones or remotely piloted vehicles (RPVs) [5] are used in coordination with sensors, fences and video cameras. More particularly, we are concerned with sensors contained in posts or alert stations called unattended ground sensors (UGS).

III. MONITORING ALGORITHM

We have proposed and implemented an algorithm by which two UAVs monitor a number of alert sites. In our simulations, values expressed in terms of percentage for different types of events (detected by a UGS) are extracted from a text file. The percentages represent the probability of occurrence of each type of event for a particular site. In addition, there is intelligence data file containing coefficient values for the same types of events. The intelligence data indicates the general pattern of events detected by a station over a longer period of time. The following steps are then performed:

1. The probability data is updated using the intelligence data.

2. For each site (for a particular timestamp), two values are computed: gain and cost.

3. For each site (for a particular timestamp), a utility value is computed where *utility = gain - cost*.

4. We apply the principle of Maximizing Expected Utility (MEU) to choose two sites, marked as active, which would be serviced by the two UAVs.

An UAV is programmed such that it will “greedily" choose an active alert station closest to their starting location or alert site.

A. Terminonology

Decision making can be defined as a process in which a selection is made from a set of alternatives or options. The selection is made in such a way so as to minimize or maximize an objective. In the rational model of decision making [6], all aspects of the process such as the set of choices, results and decision criteria are known beforehand.

A decision tree is a decision analysis tool in which a problem is represented as a directed acyclic graph with nodes and arcs [7]. A node is an element in which either a decision is evaluated or an uncertainty is calculated. The alternative or branch which gives the best overall value is the best path within the tree. A decision tree, if arranged from left to right, means that the events on the left have occurred earlier than the one on its right and so on. A decision tree is useful in graphically displaying the decision alternatives of a problem. But if there are too many variables – decision alternatives and uncertainties – the graph can quickly become large and complex. In such cases, we can represent the same problem with influence diagrams because they focus on relationship among various elements and less on statistical values.

An influence diagram (ID) is a graphical representation of a decision situation. The graph consists of nodes representing various values and arcs that depict relationships between the nodes [8]. An ID is a directed acyclic graph. It describes a decision problem in three levels: relational, functional and numerical. The relational level is the actual graphical representation of a problem using nodes and arcs that connect one node to another. The arcs define relationships or dependencies among different nodes. The functional level specifies the actual function that describes the dependencies indicated by the graphical structure at the relational level. The numerical level specifies the numerical values related to probability and utility functions. IDs were described by [9] as modeling tools to visualize relationships among random variables and resulting decisions.

B. Probability Data

An Unattended Ground Sensor (UGS) is a group of sensors that contains, but is not limited to thermal and seismic sensors. Each UGS detects events that are an aggregation from all its sensors. The events detected are classified into an event space based on a perception of threat posed by some detection. We have defined five types of events that can be sensed by a UGS. These event types also referred to as event space are: No Event (N), Small Animal (S), Large Animal (L), Human (H) and Vehicle (V). To each of these events we have assigned an importance value or weight that is totally ordered:

$$N < S < L < H < V \tag{1}$$

So, if our event space (E) is the set {N, S, L, H, V}, then we can define the weight space (W) as follows:

$$W : E \rightarrow \Re^{+|E|} \tag{2}$$

As an example for a border patrolling operation, we can say that events categorized as being human are less potent as threats than those categorized as being vehicles: W(N) = 1,W(S) = 2,W(L) = 3,W(H) = 4,W(V) = 5.

The final result of classification is a probabilistic distribution, expressed in percentage, over the event space. An example of probability distribution for an alert site, where N = no event, S = small animal, L = large animal, H = human and V = vehicle from the event space, can be: (5% N, 10% S, 20% L, 60% H, 5% V). For each alert station, one such distribution is generated for a timestamp. Therefore, there would be more than one block of data (representing all alert stations) for each timestamp value.

The probability data for a given timestamp must adhere to the following two rules:

1. In every row, the values corresponding to all the events must be ≥ 0.0.

2. The sum of the values for a site must be equal to 100. Each row must then have a sum equal to 100%.

If any one or both of the above rules are violated during error checking, the entire timestamp data is ignored and the processing moves forward to the next available timestamp.

Intelligence data is a set of coefficients over the same event space that compounds the probability distribution to either strengthen or weaken some values. In general, our assumption is that the intelligence data changes very slowly compared to its

probabilistic counterpart. Hence, intelligence is what is known to be a pattern over a longer period of time. As an analogy, we can compare probability values to weather and intelligence values to climate of any geographical region. The intelligence data for a given timestamp must abide by one rule mentioned next, otherwise it is ignored by the algorithm: In every row, each value that corresponds to an event type must be $\geqslant 0.0$.

In our decision making process, we update the probabilistic distribution values with intelligence coefficients, if they are available. Then, decision making is performed on the updated probability values. If no intelligence is known, the probabilistic distribution remains unchanged. Otherwise, intelligence data changes the probability. In other words, if something is more probable based on past experience, its probabilistic distribution is increased. For example, if an alert site is known for wild animal crossings, then the values for small and large animals are increased.

The process of updating is performed by multiplying the values (of the event space) obtained from the probability distribution with the corresponding values in the intelligence coefficients. The multiplied values are normalized to obtain final values. Consider the aforementioned probability distribution for a site given by (5, 10, 20, 60, 5). If our intelligence coefficients are (1/2, 2, 2, 2, 1, 1/2), the probability of N and V is halved, that of S and L are doubled and that of H remains the same. The calculations are performed as shown: $(5,10,20,60,5)\times\left(\frac{1}{2},2,2,2,1,\frac{1}{2}\right)=\left(\frac{5}{2},20,40,60,\frac{5}{2}\right)$. The sum $\frac{5}{2}+20+40+60+\frac{5}{2}=125$. Thus these values are normalized as:

$$\left(\frac{5}{2}\times\frac{100}{125},20\times\frac{100}{125},40\times\frac{100}{125},60\times\frac{100}{125},\frac{5}{2}\times\frac{100}{125}\right)$$

= (2, 16, 32, 48, 2) are the final updated distribution values for a site.

Ground truth is the actual event that occurred during a specific timestamp. Because the alert stations in our case are unsupervised, they do not have access to the ground truth. Therefore, we consider the ground truth to be the most probable value for each station (for a specific timestamp) breaking any ties by order (whichever comes last).

C. States of a sensing unit (UGS or UAV)

A UGS can be in either of the following states:

1. Idle: this state is denoted by I and indicates that the UGS is inactive or idle. A UGS starts in this state.

2. Alert: this state is denoted by A and indicates that the UGS has triggered an alert.

3. Service: this state is denoted by S and indicates that the UGS that triggered an alert that is being serviced by a UAV, i.e. the UAV is dwelling over or around the UGS location.

4. Undecided: this state is denoted by U and indicates that a UGS is currently inactive.

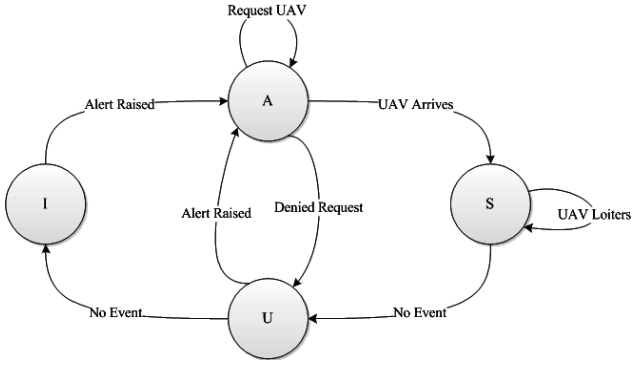

Fig. 1: States of a UGS

Once an alert is triggered, a request for a UAV is continuously generated until either the site is serviced (by a UAV) or the request is denied (for example by a control system or algorithm because the request has timed out). Additionally, once a site is serviced and there are no further events, the UGS transitions to the undecided state. From this state, it can either move to the idle state or the alert state, whichever occurs first.

A UAV can be in either of the following states:

1. Idle: this state is denoted by I and indicates that the UAV is simply patrolling.

2. Service: this state is denoted by S and indicates that the UAV is dwelling over or around a location.

3. Continue Direction: this state is denoted by CD and indicates that a specific UAV will continue in its current direction to move to a new station/site.

4. Reverse Direction: this state is denoted by RD and indicates that a specific UAV will reverse in its current direction to move to a new station/site.

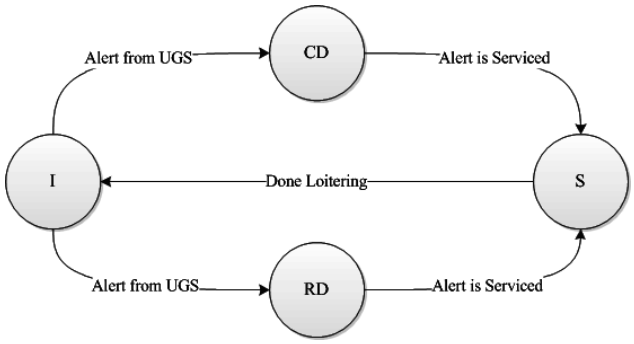

Fig. 2: States of a UAV

Two UAVs initially loiter over the two extreme edges, i.e. the first and the last alert station. Once a UAV is done servicing an alert, it would then move back to the idle state which could indicate that it is either loitering in the same station or it had moved back to its initial location/position or it is in a brief transition state before it moves to a new location for servicing. We have assumed the border to be a long straight line. That is why a specific UAV would be traveling in a straight line and would either continue or reverse its current direction to change its alert station location.

Each UAV has to move a fix number of units. The direction of the UAV, either positive or negative or zero, plays an important role in calculating these number of units. The basic steps for calculating the next direction for a UAV are given. Let the site IDs be unique consecutive numerical values starting from 1 assigned to each alert site/station (the leftmost station's ID is 1). Let p = previous Site ID of a UAV, c = current Site ID of the UAV and d = next direction value; d takes values in the set {-1, 0, 1}. Based on the values of d, each UAV could either continue its direction or reverse its direction or loiter in the same location.

D. *Value Functions*

We introduce some terms as follows.

Probability of Events (p) = $(p_j)_i$ where j is the number of event types and i is the total number of alert stations, is defined as

$$p : E \rightarrow [0,1] \quad (3)$$

Intelligence data (I) is defined as

$$I : E \rightarrow \Re^{+} \quad (4)$$

The Gain matrix (G) is a |E|x|E| matrix that measures how observed events (as reported by UGS) are compared to the ground truth (the actual event type).

Value functions are the functions that are computed during the decision making process by our algorithm.

The Gain Function (GF) of sensed data is a measure of how much a system gains by being at that specific location. Gain is based on the weight of the events and the gain matrix combined with the probability distribution and if available the intelligence data. Hence, the gain function can be defined as:

$$GF = (p, I, W, GM) \quad (5)$$

The Cost Function (CF) is the cost of not using a UAV in other alert stations when it is loitering over a given station. It is the maximum value of gain at all other stations except the current station. Hence, cost function can be defined as:

$$CF = \max(\text{gain of all sites other than the current one}) \quad (6)$$

Utility function is the numerical value of benefit obtained by engaging a UAV at a specific alert site:

$$\text{Utility} = (\text{gain} - \text{cost}) \text{ for that site} \quad (7)$$

E. *Decision Making using Influence Diagrams*

The influence diagram of the decision algorithm that we have implemented is shown in Fig. 3.

Decision making is performed in two parts. The first part consists of evaluating the function values at the individual alert stations. In the second part these values are compared in a centralized manner. The central system can be a server which operates on input information and sends commands to each UAV about their next location. Although the utility values are calculated at each node/alert station and is distributed, the actual decision making is centralized.

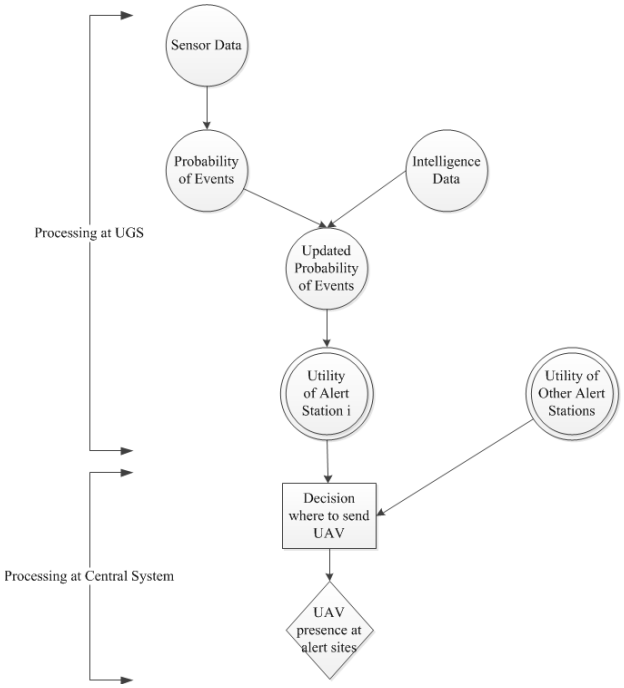

Fig. 3: Influence diagram used for our monitoring algorithm

Part 1: Processing at Alert Station i

1. In this part, the data read from the sensors is probabilistically categorized into five event types. Once categorized, this data forms the probability of events and is updated using the intelligence data's coefficients. A utility value, for every timestamp that has been recorded, is calculated following a series of steps from the probabilistic distribution data.

2. The above step is carried out in all alert stations to obtain a collection of utility values.

Part 2: Processing at Central System

1. The central system takes all the utility values from all stations and applies MEU principle to choose where to send each of the two UAVs. The decisions that are made are to send UAV1 to site x, to send UAV2 to site y and not to send the UAVs to the remaining sites.

2. The final value is the presence of the two UAVs at specific alert stations.

IV. Software Implementation and Simulation Results

The project code is implemented in the Java programming language. The Integrated Development Environment (IDE) employed for project development is Eclipse (version: Kepler Service Release 1). The code has been version controlled using the TortoiseSVN tool integrated to the IDE itself. The source code for the project is divided into two categories or from an IDE point of view into two source packages:

1. mainPackage: this package contains the core logic classes including the MainClass class to start running the project.

2. drawingPackage: this package contains all the component classes that are concerned with displaying the graphical panel of the project.

Additionally, we have used two text files as input to process the probabilistic distribution and intelligence data's coefficients. During the run of the project, a ground truth file is also created.

We have used an incremental approach to code our project. The project has been implemented in successive phases with each phase adding to or upgrading capabilities or flexibilities in solving the problem more effectively.

Single UAV - Multiple Alert Stations: The implementation started by using a single UAV to patrol all the alert stations. In this phase, we calculate a utility value for all stations but choose only one value at the end: the maximum utility. The UAV would be sent for patrolling to the corresponding site.

Two UAVs - Multiple Alert Stations with Fixed Area of Surveillance: The implementation in the first phase is extended in this phase by adding another UAV to the patrolling operations. This would again involve calculating the utility values. One station with the best utility is selected for patrolling and one of the UAVs is sent to the location. A restriction of this algorithm was that each UAV would be assigned only half or approximately half the stations for monitoring: for UAV1 the number of sites = integer(total number of sites / 2) and the rest of the sites to UAV2. There would be a boundary between patrolling areas of each UAV. Each UAV would cater to an alert only if it falls within its monitoring range.

Two UAVs - Multiple Alert Stations with Changing Area of Surveillance: The fixed boundary limitation in the second phase is removed and replaced with a moveable boundary that allows for each UAV to cater to alerts in almost any station. The only restriction is that while the UAVs can move from one active alert station to another, they do not cross-over. Each UAV has to choose the active alert station that is closest to its initial position (which is the first station for UAV1 and last station for UAV2). Hence, in every timestamp, two stations with two highest utility values are chosen. Currently, the order of choice is that UAV1 chooses its next station for patrolling which is followed by a choice by UAV2. The reverse can be done without affecting the final result. Because each UAV "greedily" chooses an active alert site closest to its starting position, the process is analogous to stretching an elastic material from a fixed point.

The *cost of a site* is the loss of all other sites not to be serviced by a UAV. If a UAV services a site, it cannot service the remaining sites. So, the cost of any site *i* is the maximum loss of not servicing a site other than *i*. In other words, the cost of site *i* is the maximum gain of all the sites other than *i*.

Per Maximizing Utility (MEU) principle, we select the utility that maximizes our benefit. Since we have two UAVs, we therefore select the two highest values from a set of utilities (for a given timestamp). These are the sites to which the two UAVs are commanded to move for monitoring in the current timestamp. Once the two best sites for monitoring are decided, the next step to be performed is to determine which UAV goes to which of the two active alert sites. While the UAVs are being positioned over the two alert stations with the highest utility values, we also calculate other gain and cost values by reading from the ground truth file. The values are the actual gain and the actual cost of moving one UAV to site I and the other to site II. These values are solely for the purpose of comparison with the estimated gain and cost values and do not currently form a feedback into the algorithm.

We have created a simple graphical panel using Java 2D. Two UAVs start at the extremes of the border line where two alert stations are kept. The timestamp at the start is zero and there are no alerts yet. This situation is shown in Fig. 4.

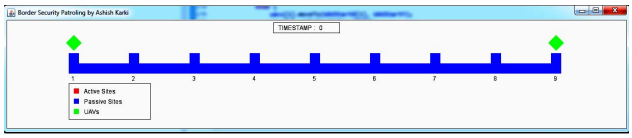

Fig. 4: Graphical Display at timestamp = 0

The project requires at least one text file as an input containing the probabilistic distribution values. Another text file containing the intelligence data's coefficients can be provided as an additional input. A ground truth file is created once the probabilistic distribution file is done reading.

Our software program reads the probabilistic distribution file and intelligence coefficients file. If any data block for a specific timestamp does not conform to the rules of validity mentioned in Section 3, for probability and intelligence data respectively, then an error message is printed out in the eclipse IDE console and the processing moves on to the next available data block. If intelligence data is available and is valid, an update of probability values is performed based on the former values. Once the values are updated, they are normalized to obtain final values.

Each site has a gain value associated with it. The calculation of site gain is based on probability values, intelligence data's coefficients, weight of the events and the gain matrix. A gain matrix is a |E|x|E| matrix with the values in the rows representing observed events and that on the columns representing ground truth. Each value represents a gain made any site for the claimed or observed event versus the actual ground truth. The values across the main diagonal (from top-left to bottom-right) are the weight of events. All the other values are relative to these weights and represents how much we would lose by wrongly categorizing an event. The gain matrix used in our calculations in shown in Fig. 5.

Ground Truth

	N	S	L	H	V
N	1	1.5	2.5	2	1
S	1	2	2.5	3	2
L	1	1.5	3	3.5	3
H	1	1.5	2	4	4.5
V	1	1.5	2	3.5	5

Observed Event

Fig. 5: Gain Matrix used in Decision Making

To calculate the gain values for each alert station, we multiply the corresponding event types from an updated probabilistic distribution and each row of the gain matrix and sum the multiplied results. The same process is repeated for each row of the matrix in order to obtain partial sums. These sums are representative of gains made if the observed event is any one of the types from the event space. All the partial sums are added and divided by the total number of events to get the final result.

As mentioned, the cost of a site is the loss of all other sites not to be serviced by a UAV and the utility is the target value obtained after all our calculations.

Error checking for the validity of data is an important part of our algorithm. If the probability values at some timestamp are deemed erroneous, they will be ignored and the data for the next timestamp is considered. Let us assume that the data for timestamps 10 and 30 are invalid. Then this data will be ignored. The next valid data is for timestamp 70. The active alert sites in this case are sites 7 and 8. These sites are marked by a red rectangle (as opposed to a blue rectangle used to mark inactive or passive sites). The UAVs are commanded to move to these sites. This situation is shown in Fig. 6.

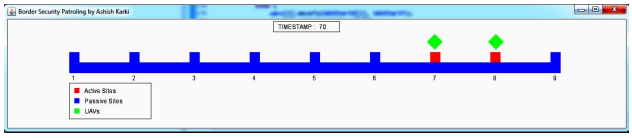

Fig. 6: Graphical Display at timestamp = 70

In the next valid timestamp which happens to be 90, the active sites/stations are 2 and 9. UAV1 is to travel to site 2 (as it is closer to its initial location which is site 1) and UAV2 is to travel to site 9 (as it is closer to its initial location which is site 9 itself). Fig. 7 shows the process of UAVs travelling to each these locations.

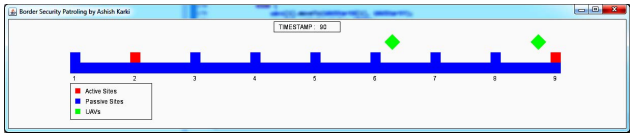

Fig. 7: Graphical Display at timestamp = 90

V. CONCLUSION

In this paper we present concepts from automatic decision making such as influence diagrams to build an algorithm which solves a decision problem. The problem is deciding where to send each UAV to dwell. This decision is based on determining two highest utility values from a set of values. And these values, in turn, are calculated for each alert stations/alert sites/UGSs from the updated probabilistic distributions. This process of calculation and decision is repeated for each valid timestamp data block that is read from the probabilistic distribution file. It is important to remember that the calculation of utility is performed at each UGS. A central system consisting of a computer server backed by one or more human operators collects all utility values and decides, using some criterion, the two sites chosen for monitoring during a specific timestamp.

Thus, utilizing an algorithm or computer program in border patrolling using UAVs and UGSs makes the overall process independent of human decision. Consequently, the decision making becomes less prone to human error and hence more efficient in terms of human resources.

The most important enhancement that can be done in the project is to make the scenario more realistic by considering the border as a region of overlapping areas. For example: when a UAV is servicing area *i*, it can also service area *i*-1 and *i*+1 simultaneously. Also, more than two UAVs can be integrated into the algorithm especially when a larger area or border line is under consideration. Within the algorithm's implementation, a restriction placed on UAVs that they cannot cross-over can be removed. This will increase the flexibility during operation and probably result in better response time to alerts.

ACKNOWLEDGMENT

The work of author Wolfgang Bein was supported by National Science Foundation grant IIA 1427584.

REFERENCES

[1] K. J. Ordonez, “Modeling the U.S. Border Patrol Tucson Sector for the Deployment and Operations of Border Security Forces”, PhD thesis, Monterey, California, Naval Postgraduate School, 2006

[2] R. W. Beard, T. W. McLain, D. B. Nelson, D. Kingston, and D. Johanson, “Decentralized Cooperative Aerial Surveillance Using Fixed-Wing Miniature UAVs," Proceedings of the IEEE, vol. 94, no. 7, pp. 1306-1324, 2006.

[3] K. Kalyanam, P. Chandler, M. Pachter, and S. Darbha, “Optimization of Perimeter Patrol Operations Using Unmanned Aerial Vehicles," Journal of Guidance, Control, and Dynamics, vol. 35, no. 2, pp. 434-441, 2012.

[4] A. R. Girard, A. S. Howell, and J. K. Hedrick, “Border Patrol and Surveillance Missions using Multiple Unmanned Air Vehicles," in Decision and Control, 2004. CDC. 43rd IEEE Conference on, vol. 1, pp. 620-625, IEEE, 2004.

[5] J. Blazakie, “Border Security and Unmanned Aerial Vehicles," DTIC Document, 2004.

[6] F. Eisenfuehr, M. Weber, and T. Langer, “Rational Decision Making”, Springer, 2010.

[7] S. Eriksen and L. R. Keller, “Decision Trees," Kluwer Academic Publishers 2001.

[8] R. D. Shachter, “Probabilistic Inference and Influence Diagrams," Operations Research, vol. 36, no. 4, pp. 589-604, 1988.

[9] A. C. Miller III, M. W. Merkhofer, R. A. Howard, J. E. Matheson, and T. R. Rice, “Development of Automated Aids for Decision Analysis," tech. rep., DTIC Document, 1976.