# UAV Search Strategies Using Cell-DEVS

**Keith Holman** keith@sce.carleton.ca
**Jeremy Kuzub** jkuzub@sce.carleton.ca
**Gabriel Wainer** gwainer@sce.carleton.ca

**Department of Systems and Computer Engineering. Carleton University**
**1125 Colonel By Drive, Ottawa, ON, Canada.**

**Keywords**: Cell-DEVS, path planning, uninhabited aerial vehicles, UAV, simulation, CD++

**Abstract**

Cell-DEVS is an extension to the DEVS formalism that allows the definition of cellular models. CD++ is a modeling and simulation tool that implements DEVS and Cell-DEVS formalisms. The methodology proposed in this paper uses the Cell-DEVS formalism and C++ tool chain [4, 5] to model Uninhabited Aerial Vehicle search in a dynamic intelligence environment in Algorithms proposed in previous works [1] were applied to modify the intelligence environment over time and to determine the UAV search pattern based on this information. The resulting simulation demonstrates emergent UAV search patterns of this intelligence environment. Rule set models the degradation of this intelligence over time using algorithms proposed in [1], referred to as a diffusion algorithm. The UAV traversed this map using a hill-climbing algorithm. The UAV searched the local maximum of target location probability before total maximum to produce an intuitive, search pattern. The Cell-DEVS architecture and CD++ tool chain provided a robust development and visualization environment suited to this research.

## 1. INTRODUCTION

UAVs provide advantages over manned aircraft for reconnaissance missions. It is advantageous for UAVs to be able to change flight plans dynamically based on new intelligence and sensor data. The methodology proposed in this paper models this changing intelligence environment as a Cellular Automata digital map. Rule set models the degradation of this intelligence over time using algorithms proposed in [1], referred to as a diffusion algorithm. A second rule set defines an efficient search pattern for the UAV using this map combined with a hill-climbing algorithm. This proposed solution was implemented in a CD++ development environment, which implemented the Cell-DEVS (Discrete Events Systems) formal specification [4, 5]. DEVS provides a formal framework for hierarchical construction of discrete-event models in a modular manner, allowing for model re-use and reduced development time. These models can be synchronous, concurrent, and highly non-linear nature. Cell-DEVS [3] was defined as a combination of cellular automata and DEVS. The CD++ tool is an open-source framework that enables the simulation of a variety of discrete-event models in biology, physics, chemistry, and artificial systems. CD++ includes a tool chain with a scripting language, simulation engine, testing environment, and graphical interface.

This paper describes an application of Cell-DEVS to the problem of finding an efficient Uninhabited Aerial Vehicle (UAV) searching pattern in a dynamic environment. The Cell-DEVS formalism provides a framework in which to simulate cellular models as discrete-event systems. This paper demonstrates how efficient Uninhabited Aerial Vehicle (UAV) search patterns can be generated by modeling target probabilities as gradients in Cell-DEVS.

This paper demonstrates that Cell-DEVS provides an intuitive and effective way to encode this information and generate dynamic path plans for UAVs. We show the results obtained when testing this model in a multi-UAV scenario and validating the path planner's effectiveness with multiple UAVs searching the same area.

## 2. BACKGROUND

UAVs provide advantages over manned aircraft for reconnaissance missions. A long loiter time allows extended search patterns and target search. During this extended mission period, many variables can change, including other UAVs in the area, target position uncertainty, new targets and points of interest, and updated intelligence.

It is advantageous for UAVs to be able to change flight plans dynamically. Traditionally this has been done by ground based operators, who in response to new information, update, waypoints and assign predetermined search patterns. In an environment with multiple targets, UAVs and prohibited zones, this coordination and manual control requires significant operator input and workload [7]. Because of the increasing capability of UAV sensors to gather large volumes of real-time data, it becomes desirable to reduce operator workload wherever possible, allowing the operator to focus on pattern recognition and sensor monitoring tasks which are best managed by human oversight.

One area in which operator workload can be reduced is by offloading is UAV path planning to automated systems. In this situation the operator assigns priorities to new intelligence and sensor information, and allows the UAV automated system to determine a search path using the fused information. While the specific algorithms for fusing data and intelligence is outside the scope of this paper, we show how the fused data can be represented as a gradient field that affects UAV search patterns.

Work by Steils and Glickstein [2] have presented the use of Cellular Automata (CA) in digital map-based tasks. Their two major applications were devoted to study situation assessment and flight path routing. They consider the cost function of the UAV movement in a terrain consisting of targets and hazards, and define methods of mapping UAV sensor input into a CA model of this environment. They showed that CA could be applied to taking many sources of intelligence and quickly and efficiently combining them to establish probable target location.

Cooperation between UAVs and assignment of priorities to determine paths have been approached from a systems level, typically with agent behaviours, and defined UAV intercommunication schemes [6, 8]. Shem, Mazzuchi, and Sarkani [1] presented a framework for fusing data sources into a CA model of a UAV's environment. They use the model to determine the probabilities of target occupancy in any given cell, which is mapped to a geographic coordinate. The UAV's path was determined by what the authors termed a 'virtual potential field'. The UAV moved in the direction of highest local target probability. The concept of prediction updating using diffusion equations was also demonstrated in the paper. This modeled the increasing uncertainty in a target's location over time due to movement between sensor detections.

By using a CA, the path planning problem can be mapped to a gradient hill-climbing algorithm in which the UAV's path is always in the direction of greatest increase in gradient. This gradient field represents probability of locating a target, and in practice, it would be a fusion of many data and intelligence sources.

Our approach is a pure cellular model, with the UAV's location embedded into the cellular model. Embedding the UAV has the benefit of allowing interaction between the UAV and the target location probability map. The UAV's presence in a given cell reduces its probability of containing the target to zero. Reducing the desirability of a cell to zero causes neighbouring cells to have a higher desirability and thereby generate a search pattern.

The research in this paper is an extension of the model first proposed in [1], which significantly reduces the complexity of planning and communication while still generating dynamic paths plans and scalable UAV cooperation implicitly. Our model simplifies the UAV path-planning algorithm development and testing by using the implementation framework of Cell-DEVS. This approach leverages the testing and visualization interfaces of Cell-DEVS and the atomic and coupled model representations of individual cell and cell-layer rules. In addition, behaviour rules for each cell layer are defined in text-based script files and different influences on UAV path planning can be isolated in separate cell layers for reconfiguration and sharing with other models.

In Cell-DEVS, each cell's value at a given time is defined by a neighbourhood of adjacent cell values via a rule set. Cell-DEVS provides a framework for discrete event simulation of these cellular models, including time and propagation delays. A tool chain for modeling and simulation using Cell-DEVS is CD++ [4], which includes a modelling language that defines cellular models rule sets and allows automated coupling of these atomic models.

The proposed model of UAV path planning involves navigating through a dynamic gradient field. This field reflects the probability of target presence versus potential danger. By summing the influences from these multiple fields in a local neighbourhood around the UAV position, a route can be generated that efficiently searches for targets. The implementation described in this paper is based on the concepts described in [1] of gradient hill climbing for UAV search, and fusion of data from multiple sensor and intelligence sources into a single "target location probability terrain".

## 3. CELL-DEVS IMPLEMENTATION

The transform from the reality of a UAV terrain exploration to a Cell-DEVS model included the following assumptions and mappings:

1. The 3D cell space was divided into three 2D layers. Layer 0 stored UAV position and previous travel path. Layer 1 stored the Target Location Probability Terrain or "gradient". Level 2 stored the UAV restricted boundary information (as seen in Figure 1).
2. The cell space was assigned a dimension of 2000 meters by 2000 meters, for a coverage area for this UAV of 4km2. This size would apply to a mini-UAV, such as the ones under development for 'back pack' deployment by soldiers or rescue teams.
3. The cell size was mapped to 100 meters by dividing UAV travel speed by rate of sensor sweeps per second for a constant search altitude of 1000 meters. This allows the model to represent UAV travel at one cell per bases simulation period (1000) without skipping or loitering in a given cell.
4. The target location probability diffusion rate varies with the speed of the target. Although a target is implicitly defined by probabilities in this model, the diffusion rate reflects the speed and unpredictable movement of the target. In this case it was modeled as a ground vehicle with a speed of 10 meters per second. Target diffusion is not direction-preferential in this model as target direction or vector information is assumed to be unknown.

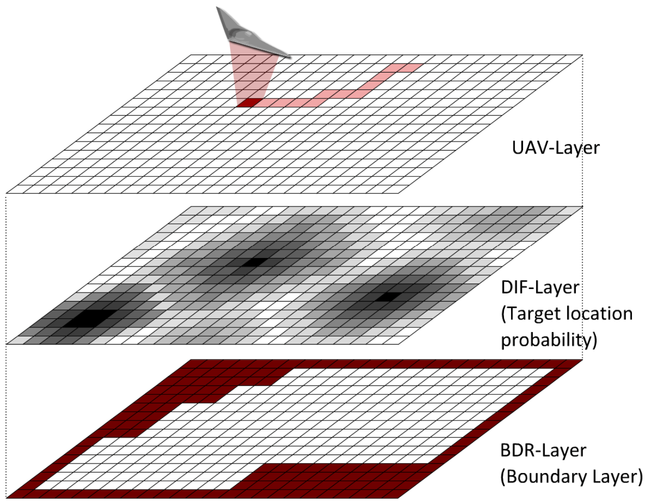

UAV-Layer
DIF-Layer (Target location probability)
BDR-Layer (Boundary Layer)

**Figure 1.** Target location probability gradient field.

### 3.1. Cell-DEVS model Rules

Each layer applies different rules that individually integrate neighbouring cells. Each layer is composed together to use additional formulas defined by Cell-DEVS that form the intelligence the UAV uses to determine its path.

The most important rule determining where the UAV will proceed is a diffusion rule. The diffusion rule models the increase of uncertainty over time for data such as target location.

In real UAV situations, initial intelligence provides a location for a target. This data becomes increasingly out of date over time due to factors like target movement. To model this effect a diffusion algorithm is implemented adds a factor of uncertainty to account for this time lag. Because the Diffusion Layer averages the probability amongst its neighbours, it takes into account a target moving into neighbouring cells and moving in any direction over time. The diffusion algorithm in implemented on its own gradient layer which allows for injection of updated intelligence. Additionally, this diffusion acts to convey information throughout the cell space, preventing a UAV from limiting its search pattern to a local maximum.

In terms of path planning, this diffusion rule provides the UAV with an effective "preference" for smaller local maxima over larger distant ones, which in practical terms means that the UAV would search for nearby targets before searching more distant ones. It also allows implicit effective cooperation between multiple UAVs searching the same gradient.

On the UAV layer, the formula of the cell containing the UAV looks at its local 8 neighbours. Out of these 8 cells, the neighbour with the maximum value is set as the direction in which the UAV will proceed. After the UAV moves to the next cell the cell value below it is set to zero. This ensures the UAV will not hover on one location.

On the Boundary layer, the value of each cell is set to a value indicating whether it’s an acceptable location to travel to. Values that are considered out of bounds are given a value of 0. Cells that are considered acceptable are given a positive value. These values are multiplied with the value of the same cell in gradient layer to effectively "mask" the gradient layer values. A cell value of 0 in the boundary layer effectively makes that direction the least interesting to UAV. This allows the algorithm to support searching an area with an irregular perimeter. In addition, this layer allows the flexibility to integrate areas that are out-of-bounds within the general vicinity.

Another benefit of this Cell-DEVS approach by defining a border on the boundary layer around the entire search region ensures that the UAV stays within the defined area. This simplifies the formulas by not requiring constant boundary checks.

The Cell-DEVS coupled model specification considers the three separate zones defined in Figure 1. These planes, are defined in order to separate rule sets for the UAV, the target location probability gradient, and the UAV path boundary. The Cell-DEVS specification provides for coupled models, which allows the model to act on a cell space neighbourhood that is a combination of these three cell spaces, or layers. This became a 3×3×3 cell space, which is specified here.

### 3.2. Cell-DEVS Conceptual Model Definition

As discussed earlier, the 3D cell space was divided into three 2D slices (or Cell-DEVS “Zones”). Layer 0 stored UAV position and previous travel path. Layer 1 stored the Target Location Probability Terrain or “gradient”. Level 2 stored the UAV “no-go” boundary information. Each zone defines a separate behaviour, as follows:

1. Zone “UAV-Layer”: this zone stores UAV's current and previous positions this information is used to move the UAV using a hill-climb algorithm with the DIF-Layer as a data source. If a UAV is in the current cell, relative values of neighbouring cells on the DIF-Layer are compared and the current cell value is changed to a ‘pending move’ value of 1-8 representing direction of pending UAV movement. Detection of a value of on the BDR-Layer is used as a flag to prohibit movement in a boundary area.
2. Zone “DIF-Layer”: the diffusion of a Target Location Probability is handled by a single rule that changes the current cells value based on an average of neighbouring cell values in the same zone. This change is slowed down by using a ‘viscosity’ divisor that can be increased to slow the diffusion rate. Diffusion has no preferential direction. An additional rule acts to reduce the target location probability of the current cell to zero if the corresponding cell the UAV-Layer Zone is occupied by a UAV. This models a UAV passing overhead and scanning the area.

3. Zone “BDR-Layer”: during research, the boundary layer remains static throughout the entire simulation; although it is feasible to use events to change the boundaries and affect the UAV path planning. A binary cell value encoding indicates if a UAV can move to the corresponding cell in its zone or is prohibited from crossing into the corresponding cell in its Zone.

## 3.3. CD++ Implementation

CD++ uses configuration files to specify the problem to be solved. This involves setting preloading cells with initial values.

Code Sample 1 shows a sample of the initial values. On the top (or UAV) layer, represented by z=0, a cell containing a UAV is set to an arbitrary value of 100.

The DIF-Layer is located on the layer with z-value of 1. For the purposes of the testing the methodology in this paper, the simulation was initialized with random values between 0 and 7. The range 0 to 7 is arbitrary and can be any value as long as the range is consistent throughout the entire field. Where 7 represents a cell with a high desirability based on the probability of a target being within that cell and 0 represents a low desirability.

The final layer, or BDR-Layer (z=2), is initialized with values of 1000 representing cells where the UAV is not permitted to go. Since this is represented as a layer rather than encoded as rules allows the boundary or "no fly zone" to be an arbitrary shape.

```
(16,12,0) = 100
(0,10,1) = 1
(0,11,1) = 1.172
(0,0,2) = 1000
(0,1,2) = 1000
```

**Code Sample 1. Subset of the configuration file that sets the initial cell values at the start of simulation**

Code Sample 2 shows static properties for the Cell-DEVS coupled model using CD++ notation. The following static information is set:

1. The model is a Cell-DEVS
2. The model should have a size of 20 × 20 × 3
3. The model uses Transport Delays
4. Borders around the model wrap around. If wrapping was not allowed then the rules for the diffusion algorithm would be more problematic around the edges because the algorithm would have to account for less neighbours contributing to the desirability. The fact that the no-fly zone layer has a border that prevents the UAV flying off the edge of the map allows for a simplification in the diffusion layer
5. The *neighbours* command defines model’s neighbourhood, which includes which cells to check if they need to be updated if one cell changes its value. The neighbourhood for the model defined in this paper is a 3 × 3 × 3 cube.
6. The *zone* attribute defines which cells are in which zones. As we can see, we split the cells into the different layers mentioned previously in this paper here. Each zone is given a name. These names relate to sections found later within the configuration file that contain rules CD++ applies to evolve the simulation.

```
[uav]
type : cell
dim : (20,20,3)
delay : inertial
border : wrapped
neighbors :  (-1,-1,0)  (-1,0,0)  (-1,1,0)
neighbors :  (0,-1,0)   (0,0,0)   (0,1,0)
neighbors :  (1,-1,0)   (1,0,0)   (1,1,0)
...
zone : UAV-layer { (0,0,0)..(19,19,0) }
zone : DIF-layer { (0,0,1)..(19,19,1) }
zone : BDR-layer { (0,0,2)..(19,19,2) }
```

**Code Sample 2. CD++ Simulation initial configuration**

Each zone defines rules in a CD++ specific notation; CD++ applies these rules during simulation execution. Each rule takes 3 parameters in the following format:

```
rule : {Postcondition} {Delay} {Precondition}
```

The *precondition* indicates a condition a cell value meet in order to trigger the rule. If the precondition is valid the cell’s value is sent to the *post condition.* The post condition specifies a new value to set. This value is spread out to neighbouring cells after a time delay. Once a cell matches a rule, then no further rules on that cell are evaluated until the next simulation step. After a cell value changes, cells in the neighbourhood are checked to see if they match any rules specific to their zones.

The direction the UAV travels in is determined by which neighbour has the highest "desirability". Code Sample 3 defines a macro in CD++ notation that calculates the highest value on the DIF-layer amongst the cells around the current location. This macro determines the neighbouring cell with the highest desirability value in the gradient field by sequentially comparing each neighbour with the highest known maximum.

```
#BeginMacro(MAX_NEIGHBOUR_1)
  max( max( max( max( max( max( max( ( (0,1,1)
- (0,1,2) ), ( (1,0,1) - (1,0,2) ) ), ( (0,-
1,1) - (0,-1,2) ) ), ( (-1,0,1) - (-1,0,2) )
), ( (1,1,1) - (1,1,2) ) ), ( (1,-1,1) - (1,-
1,2) ) ), ( (-1,-1,1) - (-1,-1,2) ) ), ( (-
1,1,1) - (-1,1,2) ) )
#EndMacro
```

**Code Sample 3. CD++ notation for defining a macro that returns neighbouring cell with the highest value offset by one layer**

The first layer, UAV-layer, is considered a zone in CD++. The model makes two passes to move the UAV. Code Sample 4 is a portion of the rule set for the first pass where the simulation determines the direction the UAV should move. This rule set changes the value of the current cell to a predetermined number that represents the direction the UAV will move. The value is only set if the current cell contains a UAV and if the neighbouring cell in the direction under test has the highest desirability amongst all neighbouring cells. The rule set also checks the boundary layer to determine if it is a valid cell for the UAV to occupy.

```
[UAV-layer]
rule : 1 1000 { (0,0,0) = 100 AND
#macro(CELL_N_2) < 1000 AND
#macro(MAX_NEIGHBOUR_1) = #macro(CELL_N_1) }
rule : 2 1000 { (0,0,0) = 100 AND
#macro(CELL_E_2) < 1000 AND
#macro(MAX_NEIGHBOUR_1) = #macro(CELL_E_1) }
...
```

**Code Sample 4.** Rule set to identify the direction to move a UAV by iterating through each neighbouring cell and determining which if that neighbour has the maximum desirability and not excluded from movement because it lies within an area that is marked out-of-bounds.

Code Sample 5 changes the current cell value to 100, which indicates the UAV is in the current cell, if the cell in the direction under test indicates the UAV was moving toward the current cell. The second last rule stipulates to set a value of 50, indicating the UAV was previously there, if the value of the cell is greater than 0.

```
rule : 100 0 { #macro(CELL_S) = 1 }
rule : 100 0 { #macro(CELL_W) = 2 }
rule : 100 0 { #macro(CELL_N) = 3 }
rule : 100 0 { #macro(CELL_E) = 4 }
rule : 100 0 { #macro(CELL_SW) = 5 }
rule : 100 0 { #macro(CELL_NW) = 6 }
rule : 100 0 { #macro(CELL_NE) = 7 }
rule : 100 0 { #macro(CELL_SE) = 8 }
rule : 50 0 { (0,0,0) > 0 }
rule : 0 0 { t }
```

**Code Sample 5.** Rule set to move the UAV if the neighbouring cell under test indicates the UAV current cell's direction.

Code Sample 6 shows the rule for the diffusion layer zone. This section contains two rules. The first rule sets the cell value under a UAV to zero. The second rule sets all other cells to a value sums that cell's current value with the average of all neighbouring cells.

```
[DIF-layer]
rule : 0 500 { (0,0,2) = 100 }
rule : { ( ( (1,0,0) + (1,1,0) + (0,1,0) + (-
1,0,0) + (0,-1,0) + (-1,1,0) + (-1,-1,0) +
(1,-1,0) ) / 8 - (0,0,0) ) / 32 + (0,0,0) }
1000 { t }
```

**Code Sample 6. Rule set for the Diffusion Layer.**

Code Sample 7 defines the rules for the Boundary Layer of the model. This rule simply keeps the value constant throughout simulation execution. However, one may expand the model in the future to support a dynamic boundary for the UAV.

```
[BDR-layer]
rule : { (0,0,0) } 10000 {t}
```

**Code Sample 7.** Rule set for the Boundary Layer, which maintains each cell's value during simulation.

## 4. SEARCH PATH SIMULATION RESULTS

The testing was divided into three stages. The first part of testing involved the rules which determined the behaviour of target location probability gradient field over time. This diffusion rule is shown in

Figure 2. The diffusion spreads high-probability 'mountains' to low probability 'valleys', modelling target location certainty reduction over time. A 'viscosity' term effectively slows down the diffusion process by modifying each cell by a fixed fraction of the difference between its value and neighbouring cell values. Testing was successful and this sub-model behaved as expected.

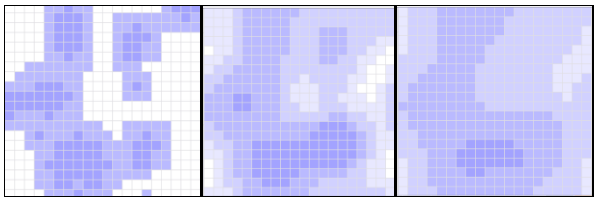


**Figure 2.** Diffusion of target location probability at three sequential times. A cell's probability of containing a target is represents by cell colour, darker colour for higher probability. Areas of Higher probability are shown to diffuse to areas of lower probability, modelling the increasing uncertainty in a target's location over time due to movement.

The second part of testing isolated the UAV hill climb rules to effectively move towards local areas of higher target location probability. The rule operates in three steps. First, a rule set determines the neighbouring cell with the highest target location probability and sets the current UAV location cell to a value which represents pending movement (1=East, 2=North etc.). The second step completes the movement by scanning neighbouring cells for a 'pending movement' value that 'points' in the current cell's direction, causing that cell to be assigned a 'UAV present' value. A final step removes residual 'pending movement' values from the zone. This hill climb was tested using a terrain generation application, which generates smooth terrain for testing the hill climbing algorithm. The diffusion rules were disabled to maintain a constant probability terrain. Viewing the results Cell-DEVS animation tool showed the hill-climbing algorithm functioned as expected (Figure 3).

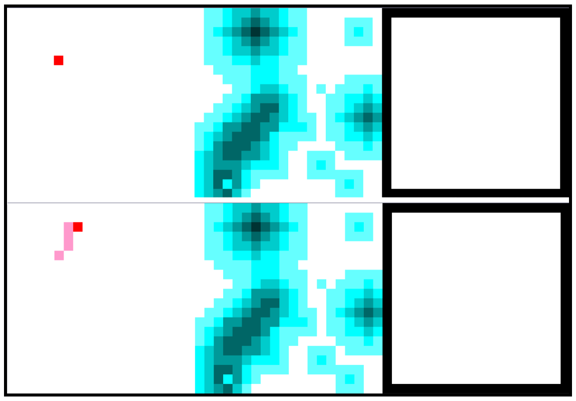


**Figure 3.** Two simulation times, top and bottom. The 3 layers of the Cell-DEVS model from left to right are the UAV search pattern, the gradient field, and the search boundaries. The UAV is shown to move towards a local maximum in the gradient field.

The third part of testing was an all up test of the target location probability diffusion rules, UAV path planning (hill climb) rules, and the go/no-go area rules. Figure 5 shows the results at three simulation steps with edge and 'island' restricted boundaries and initial terrain generated by the custom "TerrainG" tool. The second row is Simulation Step 5, showing the UAV searching the local high-probability area first; later it moved to the top right area while once-again avoiding the high probability restricted island area. The last image overlay shows that the UAV avoided the no-go areas as expected. The relatively 'flat' DIF-Layer at the end of simulation indicates that the simulation has run for some time.

In this test, all rules were active. It was tested by adding a 1-cell boundary around the cell space, eliminating and requirement for special edge detection or wrapping rules. An additional 'island' of prohibited area was added near the centre of the cell space to represent an area where flyovers are prohibited or dangerous. The UAV was then tested over an extended simulation time. No incursions took place either on the boundaries or across the restricted area. The hill climb rules for UAV operated as expected with the UAV consistently moving in the direction of greatest increase in target probability as represented in the Target Location Probability layer. Diffusion of target location probability rules operated successfully, resulting in a flatter terrain over time. The terrain was also successfully flattened to zero target probability directly 'under' the UAV as desired. The overall UAV search pattern was effective at locating local maxima of Target Probability and thoroughly searching those areas without re-crossing paths. This is relatively complex behaviour defined as simple rule set. Additionally, the Diffusion of Target Probability had the additional positive feature of the constantly providing a gradient for the UAV to 'climb', even from distant targets. This had the benefit of keeping the UAV from staying near a local maximum, leading to a thorough search of the majority of the cell area from most-to-least probable Target location areas.

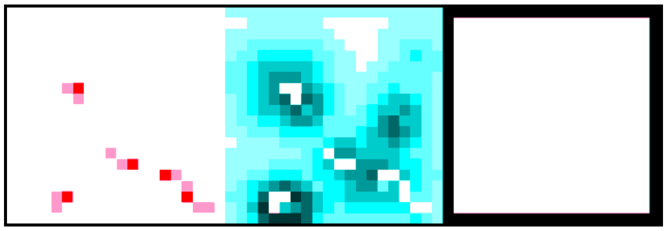
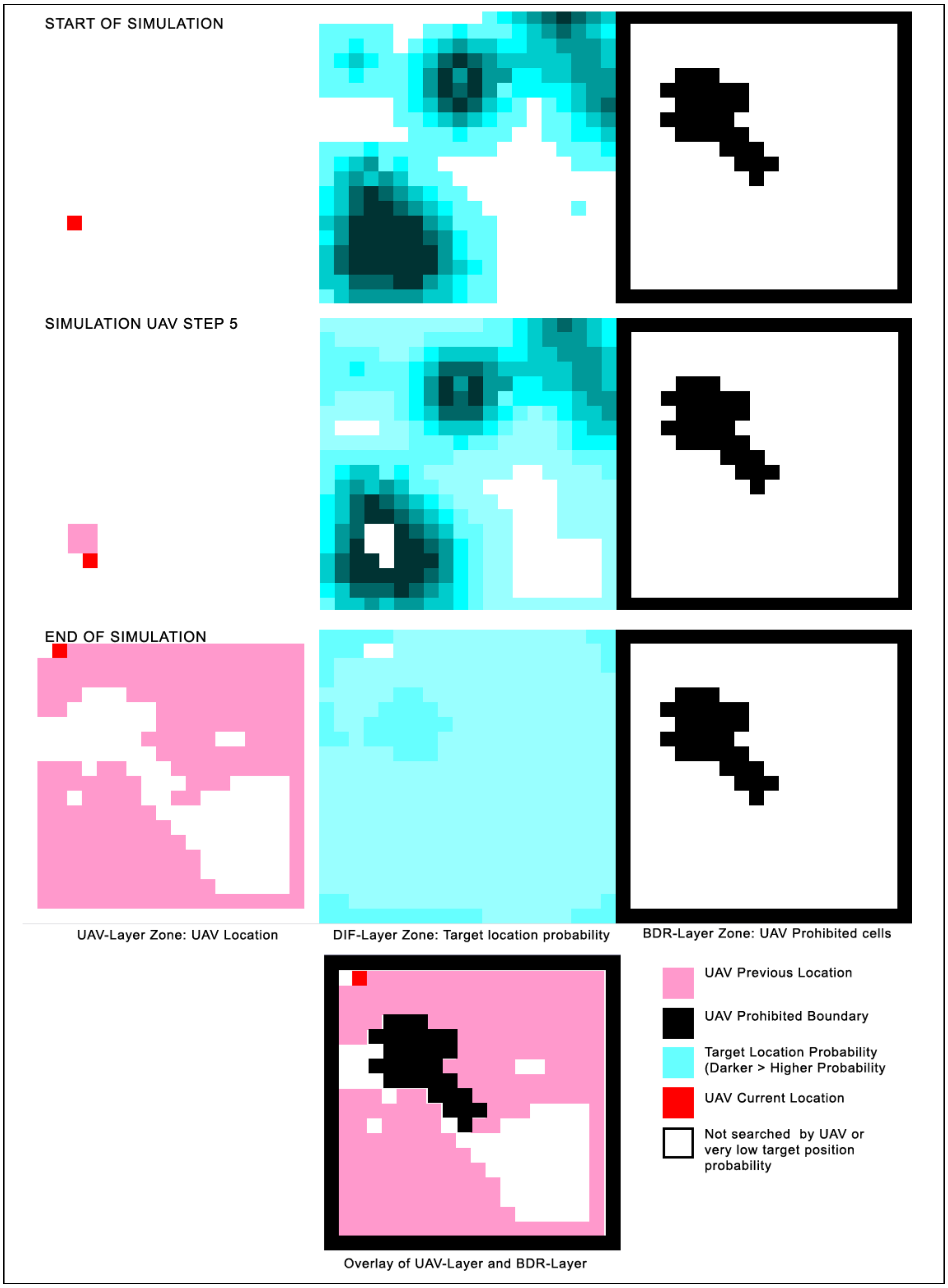


**Figure 4.** Four UAV's simultaneous search the same Target probability terrain. The three layers shown left to right are (1) the paths of the four UAVs, (2) the gradient field and (3) the search boundary**.**

Overall, the showed remarkably complex emergent UAV paths that had efficient local search behaviour, typically climbing to a local maxima then spiralling out from it to lower probabilities until the diffusion effect created a gradient to more distant maxima, which would subsequently be climbed. This architecture allows the use of multiple UAVs searching the same target location probability terrain in a coordinated fashion. There will not be repetition of search patterns since each UAV interacts with localized area and modifies the target probability terrain when passing over (

Figure 4). Additional neighbourhood patterns could be simulated and preferred target movement paths could be modeled, reflecting roads and population areas.



START OF SIMULATION
SIMULATION UAV STEP 5
END OF SIMULATION
UAV-Layer Zone: UAV Location
DIF-Layer Zone: Target location probability
BDR-Layer Zone: UAV Prohibited cells
UAV Previous Location
UAV Prohibited Boundary
Target Location Probability (Darker > Higher Probability
UAV Current Location
Not searched by UAV or very low target position probability
Overlay of UAV-Layer and BDR-Layer

**Figure 5.** Three time steps at the start, midpoint and end of a simulation as a single UAV completing a target search. The three cell layers containing information on UAV position, target location probability, and boundary areas are shown left to right. Simulation step 5 (center) demonstrates UAV initial search of a local maximum of target location probability. The last image overlays the three layers at the endpoint of the simulation, showing avoidance of boundary areas while completing search of the search zone.

## 5. CONCLUSION

The methodology proposed in this paper models a dynamic intelligence environment in which UAVs operate as a digital map in Cellular Automata. Algorithms proposed in previous works [1] were applied using the Cell-DEVS formulism. The resulting simulation demonstrated emergent UAV search patterns of this intelligence environment. Rule set models the degradation of this intelligence over time using algorithms proposed in [1], referred to as a diffusion algorithm. The UAV traversed this map using a hill-climbing algorithm. This proposed solution was implemented in a CD++ development environment, which implemented the Cell-DEVS formal specification [4, 5]. The UAV searched the local maximum of target location probability before total maximum to produce an intuitive, search pattern. A boundary cell layer was implemented to contain and limit the UAV search. The Cell-DEVS architecture and CD++ tool chain provided a robust development and visualization environment suited to this research.

## REFERENCES

[1] A. Shem; T. Mazzuchi; S. Sarkani, “Addressing Uncertainty in UAV Navigation Decision-Making”. IEEE Transactions on Aerospace and Electronic Systems Vol. 44, No. 1 January 2008

[2] I. Glickstein; P. Stiles, “Situation Assessment Using Cellular Automata Paradigm”, IEEE AES Systems Magazine, January 1992

[3] Wainer, G. 2009, “Discrete-Event Modeling and Simulation: a Practitioner’s approach”. CRC Press.

[4] Wainer, G. "CD++: a toolkit to define discrete-event models". Software, Practice and Experience. Wiley. Vol. 32, No.3. pp. 1261-1306. November 2002.

[5] J. Ameghino; G. Wainer: “Application of the Cell-DEVS formalism for modeling cell spaces” In Proceedings of AIS’2004, Jeju Island, Korea, Lecture Notes in Computer Science, 2004.

[6] F. Kamrani; R. Ayani. “Simulation-Aided Path Planning of UAV” Proceedings of the 39th Winter Simulation Conference, 2007.

[7] Hou, M.; Kobierski, R.D. : “Operational Analysis and Performance Modeling for the Control of Multiple Uninhabited Aerial Vehicles from An Airborne Platform”, Defence R&D Canada - Toronto, Toronto ONT (CAN), filename “p526642.pdf”, available online at http://pubs.drdc-rddc.gc.ca

[8] S. J. Rasmussen; P. R. Chandler: “MultiUAV: A Multiple Uav Simulation For Investigation Of Cooperative Control”, Proceedings of the 2002 Winter Simulation Conference.