# Autonomous UAV Surveillance in Complex Urban Environments

**Eduard Semsch, Michal Jakob, Dušan Pavlíček, Michal Pěchouček, David Šišlák**
Agent Technology Center, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague
Technická 2, Prague, 16627, Czech Republic
{eduard.semsch,michal.jakob,dusan.pavlicek,michal.pechoucek,david.sislak}@agents.felk.cvut.cz

**Abstract**

We address the problem of multi-UAV surveillance in complex urban environments with occlusions. The problem consists of coordinating the flight of UAVs with on-board cameras so that the coverage and recency of the information about a designated area is maximized. In contrast to the existing work, sensing constraints due to occlusions and UAV flight constraints are modeled realistically and taken into account. We propose a novel *occlusion-aware* surveillance algorithm based on a decomposition of the surveillance problem into a variant of the three-dimensional art gallery problem and the multi-traveling salesmen problem for Dubins vehicles. The algorithm is thoroughly evaluated on the high-fidelity AGENTFLY UAV simulation testbed which accurately models all constraints and effects involved. The results confirm the importance of occlusion-aware flight path planning, in particular in the case of narrow street areas and low UAV flight altitudes.

## 1. Introduction

Aerial surveillance is a task of great importance in both military and civil applications. It consists in employment of aerial assets for on-going collection of information from a specified area. Its distinguishing characteristic is the stress on recency of information and persistence. Over the last two decades there has been growing deployment of unmanned aerial vehicles (UAVs) for aerial surveillance. The present UAVs are able to perform most of their tasks autonomously and there is a strong demand for intelligent systems providing an interface for high-level commands.

A common operational picture (COP) is a military concept for materializing the situational awareness, which is however not limited to military use. The COP is a display of relevant operational information. It usually includes information about important units and infrastructure. In this paper we address the problem of obtaining and maintaining up-to-date information about the surface of the operational area using multiple *autonomous* UAVs, which we will further on refer to as *multi-UAV surveillance*. We do not deal with searching for and tracking of moving targets. The problem of multi-UAV surveillance in an urban environment is further complicated by the structure of the urban terrain. In the presence of tall buildings, the field of view of UAV sensors is likely to be *occluded*. The realistic modeling of occlusions and their explicit consideration in the proposed coordination algorithm is a distinct and novel feature of this work.

The paper proceeds as follows. In Section 2., we formalize the problem of multi-UAV surveillance in occlusion-affected environments. In Section 3., we describe our novel approach to solving the problem. The approach utilizes a discretization of the surface for which it is possible to determine a set of vantage points in the air such that any point on the surface is visible from at least one point in the set. The algorithms for generating the suitable set of vantage points and for finding a most effective path through the set is given in Section 4, including the novel *Spiral* path generation algorithm. In Section 5., we thoroughly evaluate the performance of the proposed autonomous UAV control mechanism and the improvement brought by the explicit consideration of occlusions and by the spiral algorithm. In Section 6., we briefly discuss similar approaches and provide a short comparison. We conclude with a summary and an outline of future work in Section 7.

## 2. Problem Definition

We formulate the task of obtaining and maintaining an up-to-date COP through multi-UAV surveillance as a *constrained optimization problem*. The reason is that provided the same resources, various COPs can be constructed with different but comparable quality. The quality is the recency of the information contained in COP and can be expressed quantitatively as the age of the information. In order to discriminate between more and less important pieces of information a system of priorities can be adopted.

In this section we first provide a formal description of the domain consisting of description of the environment, and characteristics of UAVs and sensors. Then we state the objective function and solution constraints and finally provide the full problem statement.

## 2.1 Model

**Environment Model** The environment in which the task is to be accomplished is an area of convex shape with surface of varying height. We assume every point of the surface can be described by its $x, y$ coordinates and a function $H(x, y)$ giving the $z$ coordinate of the surface point. More formally:

$$\begin{aligned} \text{Area} &: A \subset \mathbb{R}^2 \\ \text{Height map} &: H : A \to \mathbb{R}^+ \\ \text{Surface} &: M = \{\langle x, y, z\rangle \in \mathbb{R}^3 | \langle x, y\rangle \in A, \\ &\quad z = H(x, y)\} \end{aligned}$$

We then define the space in which the UAVs can operate (the air) as everything above the surface:

$$\text{Air} : A_{3d} = \{\langle x, y, z\rangle \in \mathbb{R}^3 | \langle x, y\rangle \in A, z > H(x, y)\}$$

We assume continuous time:

$$\text{Continuous time} : \mathrm{T} = \mathbb{R}^+$$

**UAV Model** We assume multiple UAVs and knowledge about their initial position and velocity vectors:

$$\begin{aligned} \text{UAVs} &: U = \{u_1, u_2, ..., u_n\} \\ \text{Initial state} &: I_U = \{\langle \mathbf{x}_{u_1}, \mathbf{v}_{u_1}\rangle, \langle \mathbf{x}_{u_2}, \mathbf{v}_{u_2}\rangle, \dots \\ &\quad \dots, \langle \mathbf{x}_{u_n}, \mathbf{v}_{u_n}\rangle\}, \mathbf{x}_{u_i} \in A_{3d}, \mathbf{v}_{u_i} \in \mathbb{R}^3 \end{aligned}$$

We assume a point-mass model of the UAVs and thus exclude yaw, pitch and roll angles from the initial state.

The motion of UAVs is subject to following constraints:

$$\text{Horizontal speed} : v_{h_{min}} \le \|\mathbf{v}_{xy}\| \le v_{h_{max}} \quad (1)$$

$$\text{Vertical speed} : v_{v_{min}} \le \|\mathbf{v}_z\| \le v_{v_{max}} \quad (2)$$

$$\text{Turning radius} : \arccos\left(\frac{\mathbf{v}_{xy} \cdot \dot{\mathbf{v}}_{xy}}{\|\mathbf{v}_{xy}\|\|\dot{\mathbf{v}}_{xy}\|}\right) \le \frac{v_{h_{min}}}{\rho} \quad (3)$$

$v_{h_{min}}$ and $v_{h_{max}}$ are minimal and maximal velocities the UAV can achieve when flying in one altitude. $v_{v_{min}}$ and $v_{v_{max}}$ are minimal and maximal ascending/descending velocities, and $\rho$ is the *minimal turning radius*.

**Sensor Model** Having defined the surface, air and UAVs, we move on to definition of a critical component of the problem, which is the *sensor model*. Sensor model states what part of surface is *visible* from what point in the air. We assume identical sensors and hence sensor models for all UAVs. The sensor model is given as a general function from points in the air to sets of visible points of the surface:

$$\text{Sensor model} : S : A_{3d} \to \mathscr{P}(M)$$

However, within this paper we will focus on particular sensor model of a camera with field of view having the shape of right circular cone. We assume the cone axis having direction vector $\mathbf{a} = \langle 0, 0, -1\rangle$. For this specific sensor the sensor model can be further refined. For fixed aperture of the cone $\varphi$ and a fixed point in the air $\mathbf{x}$ as its apex, we specify the visible set $S(\mathbf{x})$ as:

$$S(\mathbf{x}) = \{\mathbf{x}' \in M | \arccos\left(\frac{\mathbf{a} \cdot (\mathbf{x} - \mathbf{x}')}{\|\mathbf{a}\|\|(\mathbf{x} - \mathbf{x}')\|}\right) \le \frac{\varphi}{2} \text{ and } \forall \mathbf{y} \in C_{\mathbf{x},\mathbf{x}'} (\mathbf{y} \in A_{3d})\}$$

$C_{\mathbf{x},\mathbf{x}'}$ is the line segment connecting $\mathbf{x}$ with $\mathbf{x}'$. The first property of visible points is that they lie in the conic field of view, the second is that the line segment connecting the apex of the cone with a visible point passes only through the air.

## 2.2 Objective Function – Average Data Age

Before stating the objective function itself, we define the solution of the optimization problem to be a set of trajectories of the UAVs:

$$\begin{aligned} \text{Trajectory} &: \tau_{u_i} : \mathrm{T} \to A_{3d} \\ \text{Trajectories} &: T_U = \{\tau_{u_1}, \tau_{u_2}, \dots, \tau_{u_n}\} \end{aligned}$$

The objective function, which represents *average age of data* about the surface over a period of time, is then stated as:

$$O(T_U, t_0, t_h) = \int_{t_0}^{t_h} \left[\iint_A (t - l_M(x, y, H(x, y), t))\, dxdy\right] dt \quad (4)$$

Auxiliary function $l_M$ states for each time instance and point of surface the last previous time instance the point of surface was seen:

$$l_M : M \times \mathrm{T} \to \mathrm{T}$$

$l_M$ is calculated as follows. First we define another auxiliary function stating for a particular UAV and particular time instance the time instance it has last seen a particular point of surface:

$$Trace_{u_i} : M \times \mathrm{T} \to \mathrm{T}$$

The function can be computed as follows:

$$Trace_{u_i}(x, y, z, t) = \max\left(\{t' | \langle x, y, z\rangle \in S(\tau_{u_i}(t')) \text{ and } t' \le t\} \cup \{0\}\right)$$

Then we define the $l_M$ function for each point of surface as the maximum (latest) time when it has been seen by a UAV:

$$l_M(x, y, z, t) = \max\{Trace_{u_i}(x, y, z, t) | u_i \in U\}$$

The optimization is subject to constraints. First constraint is the initial state of the UAVs:

$$\forall u_i \left(\tau_{u_i}(t_0) = \mathbf{x}_{u_i}\right) \quad (5)$$

$$\forall u_i \left(\dot{\tau}_{u_i}(t_0) = \mathbf{v}_{u_i}\right) \quad (6)$$

Next we want the UAVs to maintain a certain relative distance $d_{safe}$ during the whole operation. We define a *deconfliction* constraint:

$$\forall t \forall u_i, u_j, i \ne j \left(\|\tau_{u_i}(t) - \tau_{u_j}(t)\| \ge d_{safe}\right) \quad (7)$$

Finally, we demand all the motion constraints of UAVs to be met.

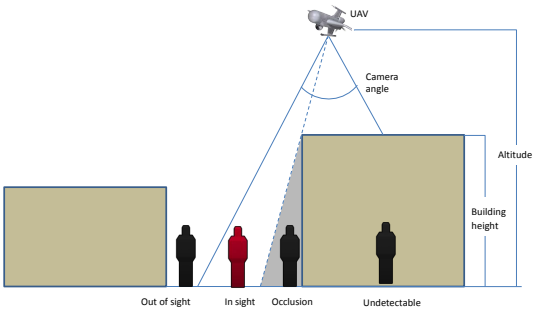


Figure 1: Occlusion-aware sensor model.

### 2.3 Problem Statement

We define the task as optimization problem: Find a set of trajectories $T_U^*$ such that

$$T_U^* = \arg\min_{T_U} \{O(T_U, t_0, t_h)\}$$

Given:

$$\langle A, H, U, v_{h_{min}}, v_{h_{max}}, v_{v_{min}}, v_{v_{max}}, \rho, \varphi, t_0, t_h, I_U \rangle$$

Subject to constraints: 1, 2, 3, 5, 6, and 7.

## 3. Approach

The problem stated in the previous section is an optimal control problem with non-linear constraints and as such it is most likely intractable. To enable an approximate solution we first make a few additional assumptions:

- All the UAVs fly in the same altitude.
- The UAVs move only in a plane, at constant speed, and along trajectories with curvature bounded by the minimal turning radius $\rho$. This simplified motion model is called the *Dubins vehicle*.
- Trajectory $\tau_{u_i}$ of a UAV is fully described by a sequence of points in the air and we assume a path-planner capable of finding the shortest path for a Dubins vehicle (termed shortest *Dubins path*) through the ordered points.
- The surface consists only of the base plane (plane defined by equation $z = 0$) and quadrangular prisms lying on the base plane.

The resulting sensor model is depicted graphically in Figure 1. Assuming the structure of the surface to be a composition of quadrangular prisms, ensures that there is such a finite set of points in the air, all lying in the same altitude, that every point on the surface can be seen from at least one of the points in the set (De Berg et al. 1997). We term this set a *covering vantage point set*.

### 3.1 Problem Decomposition

Adding the above assumptions to the problem statement enables us to decompose the problem into two subproblems that can be solved sequentially:

1. Solving an instance of the *3D art gallery problem* to produce a covering vantage point set.
2. Solving an instance of the *traveling salesman problem for a Dubins vehicle* to order the points in the covering set into an optimum path.

The *art gallery problem* (Marengoni et al. 2000) refers to finding the minimal number and positions of sensors in a polygonal area with or without polygonal holes such that any point inside the polygonal area can be seen by at least one sensor. In the basic formulation the sensors are assumed omnidirectional. The problem has been shown to be NP-hard. An often used approximation approach is to discretize the monitored area and the area where the sensors can be placed, compute the visibility between these two sets, and find a minimal set cover. While computing the minimal set cover is also a hard problem, efficient approximation algorithms exist. In our approach we have proceeded along these lines as will be discussed in the Section 4.

The *Traveling salesman problem (TSP)* is a well known optimization problem to find the shortest closed tour through a set of cities. When the cities are points in a plane and the salesman travels along Dubins paths with travel cost proportional to length of the paths, the problem is referred to as *TSP for Dubins vehicle (DTSP)* (Savla 2007). The additional restrictions imposed on the movement of the salesman render the wealth of algorithms for TSP not directly applicable and call for a specialized algorithms. We describe two of such algorithms in the next section.

## 4. Algorithms

We now describe specific algorithms used to solve the two constituent problems outlined above.

### 4.1 Covering Vantage Point Set Creation

We use the discretization-based approach to find an approximately optimal covering set of vantage point. The surveilled area is discretized into a *visibility verification grid (VV grid)* which represents the set of points the visibility of which is sought by the covering set generation algorithm. The algorithm itself proceeds in two steps:

1. For a given altitude, construct a regular grid of points in the air lying in a plane parallel to the surface. Find all surface points in the VV grid visible from the air grid. If not all points in the grid are covered, refine the grid and reiterate.
2. Use the greedy set cover algorithm to reduce the number of points in the covering vantage point set.

Because of the assumption on the surface structure, the surface can be modeled as a set of polygons which makes the visibility test in Step 1 easy to compute. The greedy set cover algorithm used in Step 2 is a polynomial-time approximation algorithm for finding a minimal set cover of a given set. At each step, the algorithm adds such a set to the cover that contains most

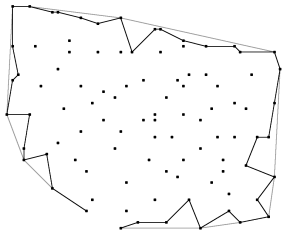

Figure 2: Convex hull (thin line) and relaxed hull (thick line) of a set of waypoints.

elements not covered up to that step. In our case, the elements correspond to points in the VV grid and the sets corresponds to sets of VV grid points visible from a single point of the air grid.

After performing the above two steps, we obtain an approximation of a minimal set of points in the air at a given altitude that together cover all surface points in the VV grid.

### 4.2 Algorithms for Dubins Vehicle Traveling Salesman Problem

Having generated a covering set of vantage points, the second step is to solve the DTSP for this set. Although some algorithms for DTSP exist, they proved less suitable for the instances of DTSP generated by the surveillance problem. We have therefore developed a novel DTSP algorithm, termed *spiral algorithm*, based on spiral-shaped flight trajectories. In addition, we briefly describe the best existing general algorithm for DTSP called *alternating algorithm*, which we used for comparison.

**Spiral Algorithm** The algorithm sets the order of the given points in a plane so that the resulting Dubins path forms a shape resembling a spiral. The algorithm is iterative. Given a set of points, the first iteration constructs a convex hull of the set and the points forming the boundary of the hull are then removed from the set. The remaining points are then used as the input for other iterations. The process is repeated until there are no points left.

The individual hulls we receive this way are continuously linked together. We start with the outer-most hull. For each pair of neighboring hulls we search the shortest possible link, i.e. a pair of points, to merge them together. Table 1 shows the pseudocode of the described algorithm.

This basic algorithm is further improved by relaxing the convex hulls (see Fig. 2 and function `RelaxConvexHull` in the pseudocode). The idea is to include as many points into each hull as possible while still keeping the hull smooth enough for the UAV to fly through it without problems. After constructing each convex hull, we inspect whether there are any points inside the hull that are close enough to the hull's edge

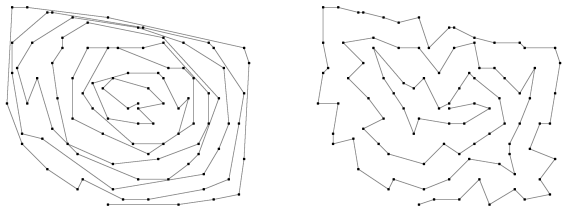

Figure 3: Spiral trajectory consisting of non-relaxed convex hulls vs. relaxed hulls. In this particular case, the length of the relaxed trajectory is only 60% of the non-relaxed one.

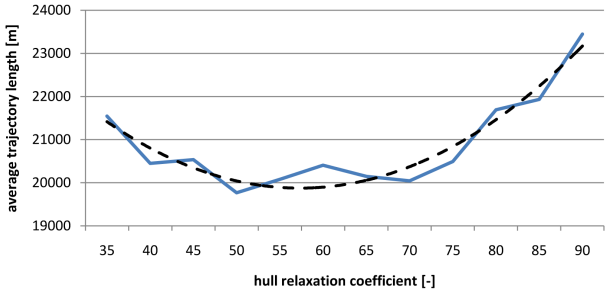


Figure 4: Influence of the convex hull relaxation on the length of the flight path.

and could be added into the (relaxed) hull without the UAV having to change its current course too much. If such a point is found (line 27 of the pseudocode), the convex hull is relaxed[1]. This relaxation process is executed recursively for each edge of the hull.

Figure 3 illustrates the influence of the relaxation of convex hulls on the length of the resulting spiral trajectory. The horizontal axis in the graph represents the *relaxation coefficient* $K$ whose relation to *angle_threshold* referred by Algorithm 1 is defined by the following formula ($C$ is a normalizing constant):

$$angle\_threshold = K \cdot \frac{edge\_length}{C}$$

The vertical axis of the graph represents the average trajectory length measured for a number of different altitudes of the UAV. Based on the graph, we have chosen 65 as a suitable relaxation coefficient.

The algorithm has approximately quadratic time complexity. Fig. 5 shows the dependency of the runtime of the spiral algorithm on the number of input points. The points lie inside a square area and they are drawn from uniform distribution. Each point of the graph is an average of 100 runs.

**Alternating algorithm** Alternating algorithm (Savla 2007) is an approximation algorithm for solving DTSP with known upper and lower bounds on solution

[1]Note that the resulting polygons are no longer convex

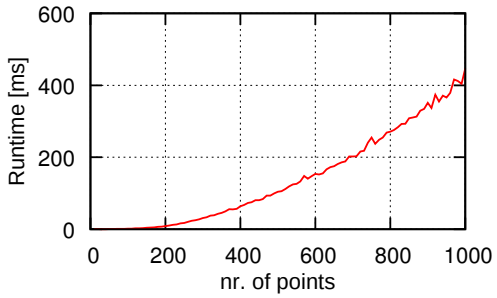


Figure 5: Runtime performance of the spiral algorithm

```
1  function BuildSpiral (points)
2  begin
3      spiral_points ← empty list;
4      while points not empty do
5          hull_points ← ConstructConvexHull (points);
6          points.Remove (hull_points);
7          RelaxConvexHull (hull_points,points);
8          spiral_points.Append (hull_points)
9      end
10     return spiral_points;
11 end
12 function RelaxConvexHull (hull_points,inner_points)
13 begin
14     changed ← true;
15     while changed = true do
16         changed ← false;
17         i ← 0;
18         while i < hull_points.Size do
19             j ← (i + 1) mod hull_points.Size;
20             edge_start ← hull_points [i];
21             edge_end ← hull_points [j];
22             nearest_point ← FindNearestPointToEdge
               (inner_points,edge_start,edge_end);
23             angle1 ← Angle (Line
               (edge_start,edge_end),Line
               (edge_start,nearest_point));
24             angle2 ← Angle (Line
               (edge_end,edge_start),Line
               (edge_end,nearest_point));
25             edge_length ← Distance
               (edge_start,edge_end);
26             angle_threshold ← K× edge_length / C;
27             if (angle1 < angle_threshold) and (angle2 <
               angle_threshold) then
28                 hull_points.InsertAt (j,nearest_point);
29                 inner_points.Remove (nearest_point);
30                 i ← i + 1 ;
31                 changed ← true;
32             end
33             i ← i + 1 ;
34         end
35     end
36 end
```

**Algorithm 1**: Pseudocode of the spiral algorithm

quality. The algorithm builds upon the fact that shortest Dubins path from one point to another can be easily found given the heading vectors (the directions in which the vehicle should pass the points) in both the points.

The algorithm works in two stages. First the ordering of points is produced using an optimal solver for *Euclidean traveling salesman problem* and then the heading vectors are calculated according to following scheme: For each odd pair of consecutive points the heading vector in both points is set to have the direction of the line segment connecting the two points and the sense from the first to the second point in the pair. If there is an odd number of points, the heading vector for the last point is directed towards the first point.

The implementation of the algorithm was straightforward. We have utilized freely available Euclidian TSP solver LINKERN [2] for the first phase of the algorithm and fed the ordered waypoints to the AGENTFLY path planner (Sislak, Volf, and Pechoucek 2009), which works similarly to the second phase of the algorithm.

### 4.3 Multiple UAVs

Note that the solution described in the previous sections is applicable to one UAV. There are two principal ways in which it can be used for multiple UAVs: (1) the surveilled area is partitioned into as many subareas as there are UAVs and each area is then handled independently by a dedicated UAV (2) each UAV surveils the whole area and the placement of UAVs is coordinated so that their spacing is optimized. In our solution, we have utilized a simple variant of the latter approach. The starting points of the UAVs are determined so that the dispersion of the UAVs over the target area is maximized. Starting from its initial position, each UAV then applies the surveillance algorithm independently without any further coordination.

## 5. Experimental Evaluation

Our objectives in the experimental evaluation were twofold. Primarily, we wanted to analyze the performance of the proposed spiral-based occlusion-aware surveillance algorithm with respect to the objective function introduced in Section 2.2 and we compared the performance to that of the alternating algorithm. Secondarily, we evaluated the extent to which the explicit consideration of occlusion improved the coverage of the surveilled area compared to occlusion non-aware algorithm.

### 5.1 Simulation Testbed

All evaluation was conducted within the AGENTFLY framework[3] (Pechoucek and Sislak 2009) for flight and air traffic simulation. The core framework – incorporating the air traffic domain model, accelerated flight

[2] http://www.tsp.gatech.edu/concorde/

[3] http://agents.felk.cvut.cz/projects/agentfly/

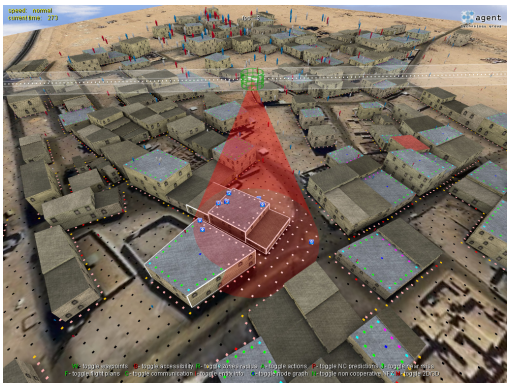

Figure 6: AgentFly UAV simulation testbed with an occlusion-aware sensor model.

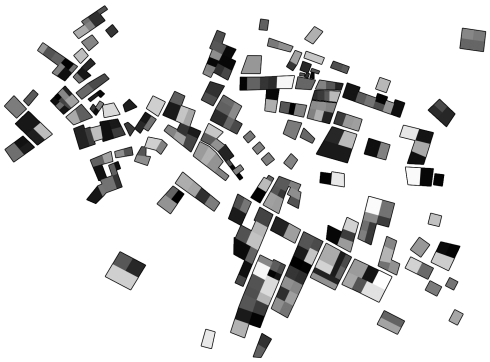

Figure 7: Visualization of the height map used for the experiments. Highest buildings depicted in very light gray (22m), lowest buildings in black (6m).

path planning (Sislak, Volf, and Pechoucek 2009) and collision avoidance – have been extended with a realistic on-board sensor model which accurately simulates the effect of occlusions. A screenshot of the simulation testbed in operation is given in Figure 6.

### 5.2 Test Scenario

The specific scenario used for the evaluation is modeled after a real-world village with surroundings located in a flat 1500m-by-1500m square area. Buildings are modeled as non-overlapping but possibly adjacent quadrilateral prisms with bases on the $z = 0$ plane. There is total of 300 buildings with heights in 6 to 22 m range; the width of the streets range from 3 to 10 meters. The whole 1500m-by-1500m village area is to be surveilled. A visualization of the height map used for the experiments is depicted in Figure 7

There are a number of configurable parameters of the scenario, summarized in Table 1 including the range within which they were varied.

| | |
|---|---|
| Number of UAVs | 1–6 |
| UAV altitude | 50–300 m |
| UAV minimal turning radius | $\rho = 0$m-40m |
| Sensor aperture angle | $\varphi = 47^\circ$ |
| UAV speed | $v_{h_{min}} = v_{h_{max}} = 25$ m/s |

Table 1: Parameters of the experiments

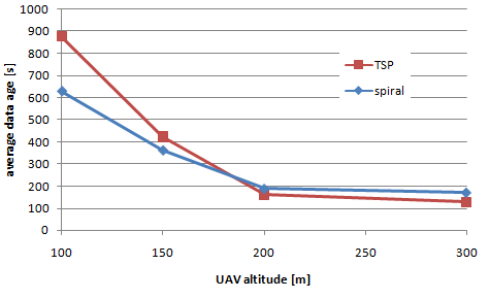


Figure 8: Average data age for the spiral and the alternating (TSP) variant of the occlusion-aware algorithm

### 5.3 Average Data Age

For evaluating the performance of the proposed surveillance algorithm, we have used a discretized version of the average data age objective function (see Equation 4) defined as

$$O_d(T_U, t_0, t_h) = \frac{1}{t_h - t_0} \sum_{i=\lfloor \frac{t_0}{T_s} \rfloor}^{\lceil \frac{t_h}{T_s} \rceil} \left( \sum_{\mathbf{x} \in V} (T_s i - l_M(\mathbf{x}, T_s i) \right) \quad (8)$$

$T_s$ is a time sampling period and $V$ is the covering vantage point set. Our primary aim was to investigate the dependence of average data age on the flight altitude and turn radius as the two parameters critically affecting occlusions and the ability of the UAV to execute complicated paths. The results for the spiral and the benchmark alternating algorithm are depicted in Figure 8, respectively.

### 5.4 Trajectory Length

In order to better understand the effect of different waypoint ordering algorithms, we have also measured the length of the trajectories followed by the UAVs during surveillance. We have again evaluated the dependence on the flight altitude and the turn radius; the results are depicted in Figure 9.

### 5.5 Relative Coverage

Due to its design, the occlusion-aware surveillance algorithm has a guaranteed 100% coverage of the target area when measured on the visibility-verification grid (VV grid). Although this does not necessarily amount to 100% of the real surface surveilled, the uncovered

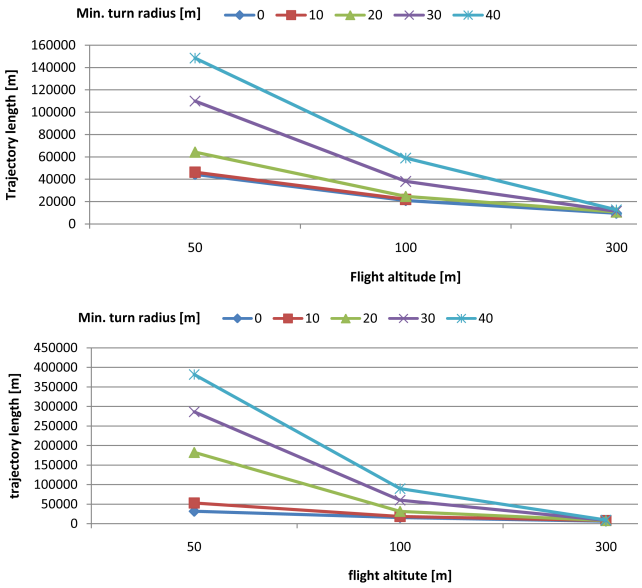


Figure 9: Trajectory length for the spiral (top) and alternating (TSP) (bottom) occlusion-aware algorithm for different altitude height and turn radius

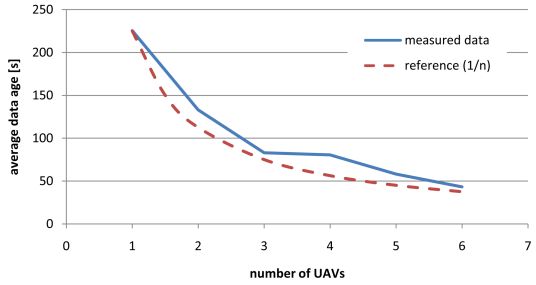


Figure 10: Average data age for the multi-UAV spiral occlusion-aware algorithm.

area can be made arbitrarily small by increasing the resolution of the VV grid. To quantify the effect of occlusion, we have applied an optimum occlusion non-aware surveillance algorithm (Huang 2001) on the test scene. The algorithm achieved approximately 95% coverage of VV grid with most of the uncovered points hidden in the narrow streets.

### 5.6 Number of UAVs

Finally, we measured the improvement in average data age when multiple UAVs were used. For that, we used the simple independent surveillance algorithm described in Section 4.3 which only required the coordination of the UAVs to happen at the beginning of the surveillance task. Results given in Figure 10 show that even without any explicit coordination during the flight, the UAVs are exploited in a surprisingly efficient way.

### 5.7 Discussion

The experiments indicate a critical importance of the UAV flight altitude on the efficiency of surveillance in occlusion-affected areas. Increasing UAV flight altitude helps both in extending the overall effective ground area covered by the sensor and in improving the area within which the sensor's line-of-sight is near orthogonal to ground, and hence unaffected by occlusions. In practice, increasing flight altitude has a negative effect on sensor resolution and can therefore be only elevated to a certain limit. In addition to the flight altitude, the UAV's minimum turn radius plays a crucial role as demonstrated by the experiments evaluating the total flight path length of each surveillance cycle.

As far as the comparison of the spiral and the alternating algorithm is concerned, the results tend in favor of the former. Except for a rather high flight altitudes, the spiral algorithm outperforms the alternating algorithm and considerably so in low flight altitudes.

## 6. Related Work

The problem of multi-UAV surveillance has received some attention lately and a variety of approaches from reactive policies to deliberative search-based methods have been proposed. However, no approach to UAV surveillance has been found that explicitly deals with occlusions.

In (Nigam and Kroo 2008) the authors present an approach to construction of a semi-heuristic control policy for multiple UAVs performing a surveillance task. In (Caffarelli et al. 2003) Caffarelli et al. propose a class of semi-distributed stochastic navigation algorithms based on minimization of artificial potentials with two aims: 1. to provide a robust and efficient algorithm for surveillance and 2. to decrease the predictability of trajectories the group of UAVs follows.

The more deliberative approaches pose the surveillance problem as a *routing problem* as e.g. in (Ryan et al. 1998) where the resulting problem is *traveling salesman problem with time windows*.

In some routing problems in the UAV domain, especially when the waypoints are near to each other, it is needed to consider the aircraft trajectory constraints. Simplest but satisfactory approximation of a fixed-wing aircraft motion model is the Dubins vehicle. A notable work on routing problems with Dubins vehicles is (Savla 2007). The work describes single and multi UAV routing problems including TSP for fixed-wing aircraft and rotorcraft and provides a set of approximation algorithms with stated upper and lower bounds on their performance. For DTSP the author introduces two novel approximation algorithms: The *alternating algorithm* that has been described in Section 4. which produces feasible approximations of the shortest Dubins path for adversarial distributions of points and the *bead-tiling algorithm* which is a constant factor optimization in cases when the points are sampled from uniform distribution.

## 7. Conclusion

We have formalized the problem of UAV surveillance in occlusion-affected environments and proposed a modular approach towards solving the problem. The approach decomposes the overall constrained-optimization surveillance problem arising form the formalization into two subproblems, which can be solved independently. We then described how these resulting subproblems can be solved and proposed a novel spiral algorithm which produces shorter surveillance trajectories in occlusion-affected environments than existing general purpose algorithms. We have evaluated the resulting set of algorithms on realistic surveillance scenario modeled and simulated using the AGENTFLY UAV simulation framework. The results show a critical importance of the UAV flight altitude and its minimum turn radius on the surveillance performance measured in terms of the average data age and surveillance trajectory length, with low turn radiuses and high altitudes having a positive effect.

Many improvements to the presented research are possible. Two of the most imminent include (1) the extension of the surveillance problem definition and the solution algorithms to take into account prioritisation of different parts of the target area and (2) more sophisticated coordination algorithms for multi-UAV surveillance. In the future, we plan to apply the developed occlusion-aware sensing model also to mobile target tracking and other UAV information gathering tasks.

## Acknowledgment

The research has been sponsored by the U.S. Army (grant no. W911NF-08-1-0521 1312AM01) and also by the Czech Ministry of Education, Youth and Sports under grant "Decision Making and Control for Manufacturing III" (grant no. MSM 6840770038).

## References

[Caffarelli et al. 2003] Caffarelli, L.; Crespi, V.; Cybenko, G.; Gamba, I.; and Rus, D. 2003. Stochastic Distributed Algorithms for Target Surveillance. *Intelligent Systems Design and Applications* 137.

[De Berg et al. 1997] De Berg, M.; Halperin, D.; Overmars, M.; and Van Kreveld, M. 1997. Sparse arrangements and the number of views of polyhedral scenes. *International Journal of Computational Geometry and Applications* 7:175–196.

[Huang 2001] Huang, W. 2001. Optimal line-sweep-based decompositions for coverage algorithms. In *IEEE International Conference on Robotics and Automation*, volume 1, 27–32. IEEE; 1999.

[Marengoni et al. 2000] Marengoni, M.; Draper, B.; Hanson, A.; and Sitaraman, R. 2000. A system to place observers on a polyhedral terrain in polynomial time. *Image and Vision Computing* 18(10):773–780.

[Nigam and Kroo 2008] Nigam, N., and Kroo, I. 2008. Persistent Surveillance Using Multiple Unmanned Air Vehicles. In *2008 IEEE Aerospace Conference*, 1–14.

[Pechoucek and Sislak 2009] Pechoucek, M., and Sislak, D. 2009. Agent-based approach to free-flight planning, control, and simulation. *IEEE Intelligent Systems* 24(1):14–17.

[Ryan et al. 1998] Ryan, J.; Bailey, T.; Moore, J.; and Carlton, W. 1998. Reactive tabu search in unmanned aerial reconnaissance simulations. In *Proceedings of the 30th conference on Winter simulation*, 873–880. IEEE Computer Society Press Los Alamitos, CA, USA.

[Savla 2007] Savla, K. 2007. *Multi UAV Systems with Motion and Communication Constraints*. Ph.D. Dissertation, UNIVERSITY of CALIFORNIA.

[Sislak, Volf, and Pechoucek 2009] Sislak, D.; Volf, P.; and Pechoucek, M. 2009. Accelerated A* Path Planning (Extended Abstract). In *(to appear) AAMAS '09: Proceedings of the eighth international conference on Autonomous agents and multiagent systems*.