UNSYS digital

Journal of Unmanned System Technology

Patrolling Strategy Using Heterogeneous Multi Agents in Urban Environments Using Visibility Clustering

Oren Gal[†], Yerach Doytsher[†]

[†]Mapping and Geo-information Engineering, Technion - Israel Institute of Technology

Haifa, Israel.

***Abstract*—** In this paper, we study the Visible Trajectories Planning for Patrolling application using heterogeneous multi agents in 3D urban environments. Our concept is based on a spatial clustering method using visibility analysis of the 3D visibility problem from viewpoints in 3D urban environments, defined as locations. We consider two kinds of agents, with different kinematic and perception capabilities. Using a simplified version of Traveling Salesman Problem (TSP), we formulate the problem as one of patrolling strategy, with upper boundary optimal performances. Our algorithm present a combination of relative deadline UniPartition approaches based on visibility clusters. These key features allow new planning for an optimal patrolling strategy for heterogeneous agents in urban environments. The patrolling strategy method is demonstrated using Autonomous Navigation and Virtual Environment Laboratory (ANVEL) test bed simulation.

Keywords—Multi Agents, Visibility, Clustering

I. INTRODUCTION

THE MULTI AGENT patrol strategy problem in urban environments deals with optimal strategy planning for heterogeneous agents patrolling along different locations for several missions, such as protection, surveillance, maintenance, etc. Although, there may be different goals for patrolling strategy (goal functions), all these missions are related to the time dimension and sensor-based 3D visible volumes involved in mission optimality.

Patrolling problem known as one [1]. Traveling Salesman Problem (TSP), can be defined as one agent and one visit for every location along the route [3]. The goal of the patrolling strategy is usually to minimize the at each location, in which the time passes between consecutive visits. In [2] another complication presented, in the form of the Relative Deadline constraint, defined as the maximum time that may pass between consecutive visits. In [2], the authors focuses on protection against terrorist acts; hence the constraint was set by the amount of time it allegedly takes a terrorist to commit an attack on a certain location based on its vulnerability. In other cases deadline can be defined as the time between malfunctions in the locations based on the reliability of each location. In cases of online data-sharing between agents, such as a distributed network based on communication limitations between the agents, patrolling for surveillance can be done by other efficient methods [4].

In [1], three different strategy approaches were suggested:

1. *Single Cycle* - Finding a single path through all the locations so that all agents will follow at a certain frequency.
2. *UniPArtition* - Dividing the locations into clusters (according to the number of agents), where each agent is given a certain cluster to patrol in. Obviously, in each cluster a Single Cycle patrolling strategy will be set. We extend the cluster method and propose 3D Visibility Clustering.
3. *Multi Partition* - Dividing the locations into clusters (while and is the number of agents) and allocating more than one agent to each cluster.

In this paper, we focus on combination of relative deadline with UniPartition approaches based on visibility clusters. These key features allow new planning for an optimal patrolling strategy for heterogeneous agents in urban environment.

II. RELATED WORK

A. Multi-Agents Systems

Multi-agents decision-making and control methods can be divided into two major disciplines: centralized and decentralized approaches. The basic idea of the centralized approach is to make all the decisions in one place. All tasks are concentrated by a single entity, named 'Central Task Planner and Scheduler' (CTPS). The CTPS translates the tasks into smaller tasks (sub-tasks), which will later be sent to the appropriate agents, according to their capabilities, their assignment and their workload. Theoretically, the centralized approach appears to fulfill its purpose. It allows advance knowledge of all the tasks to be done and the connections between them, enabling the choosing the most fitting disassembling of the problem into sub-tasks. Indeed, this is a significant advantage, as there is no disassembling which would be ideal for all missions.

However, this approach does not fit a dynamic environment, in which unpredictable events may occur. Multi-agents in a marine environment are usually not in constant contact with CTPS or with each other, even though the CTPS requires a continuous stream of data about forthcoming events in order to provide an effective response. Solutions to this problem (such as placing multiple sensors in the environment) are expensive and hard to apply.

On the other hand, with the decentralized approach, each

agent is responsible for a group of tasks, and there is no need to use an entity such as CTPS. A predetermined disassembling is applied to the problem, and the agents can try to contact each other in order to improve it. As mentioned above, this solution is problematic, as there is no disassembling which would be ideal for all problems.

Despite this fact, the lack of a CTPS allows every agent to process the data it collects by itself, and, for example, to plan its own trajectory using local sensors data and decide what the next action is. The benefit of this approach is, of course, the speed of reaction and the independence of the agents. Moreover, it allows real-time reaction to dynamic changes in the environment while a complete review of the existing work on multi-agent motion planning is a vast field, we focus here on multi-agents in a marine environment. Guo [19] presented a distributed control method with a group of robots. However, this method focused on velocity planning and optimal paths so that the robots would reach their destinations as quickly as possible, without colliding with each other. The authors do not discuss missions that are more complicated than trajectory planning (such as intelligence gathering), or the option of autonomously changing an agent's destination on a path.

An interesting work is the Cocktail Party Model, where [20] discuss the control of a group of robots without mentioning patrol or complex missions. It offers an entirely distributed approach, where there is no communication between the agents. Each agent plans and alters its destination dynamically whilst moving. The agents avoid static obstacles in the environment, as well as other agents, which are considered "moving obstacles".

This approach still does not tackle decision-making aspects other than trajectory planning [17] introduced a centralized approach that deals with complex missions that require more than one agent. The concept allows the alteration of missions after more knowledge of the environment has been accumulated, making it possible to choose a different path at a lower cost. Moreover, the presented method allocates multiple destinations per agent. However, the concept omits patrol or other complex missions other than trajectory planning. A combination of the centralized and distributed approaches is introduced by [18] . The method is based on synchronizing the movement of the agents regardless of the specific mission.

Recently, [1] introduced distribution of patrol points so that the maximum time each point is left unvisited by an agent is minimized. The authors presented a centralized approach. However, the authors do not discuss heterogeneous agent's cases.

B. Visibility Analysis in 3D Urban Environments

The visibility issue has been extensively studied over the last twenty years, due to the importance of visibility in GIS and Geomatics, computer graphics and computer vision, and robotics. Accurate visibility computation in 3D environments is a very complicated task demanding a high computational effort, which could hardly have been done in a very short time using traditional well-known visibility methods [7]. The exact visibility methods are highly complex, and cannot be used for fast applications due to their long computation time. Previous research in visibility computation has been devoted to open environments using DEM models, representing raster data in 2.5D (Polyhedral model), and do not address, or suggest solutions for, dense built-up areas. Most of these works have focused on approximate visibility computation, enabling fast results using interpolations of visibility values between points, calculating point visibility with the Line of Sight (LOS) method [8]. Other fast algorithms are based on the conservative Potentially Visible Set (PVS) [9]. These methods are not always completely accurate, as they may render hidden objects' parts as visible due to various simplifications and heuristics.

A vast number of algorithms have been suggested for speeding up the visibility process and reducing computation time. Franklin 2002, evaluates and approximates visibility for each cell in a DEM model based on greedy algorithms. Wang et al. 1996 [10], introduced a Grid-based DEM method using viewshed horizon, saving computation time based on relations between surfaces and the line of sight (LOS) method. Later on, an extended method for viewshed computation was presented, using reference planes rather than sightlines [11].

One of the most efficient methods for DEM visibility computation is based on shadow-casting routine. The routine casts shadowed volumes in the DEM, like a light bubble [12]. Extensive research treated Digital Terrain Models (DTMs) in open terrains, mainly Triangulated Irregular Network (TIN) and Regular Square Grid (RSG) structures. Visibility analysis in terrain was classified into point, line and region visibility, and several algorithms were introduced, based on horizon computation describing visibility boundaries [13].

Only a few works have treated visibility analysis in urban environments. A mathematical model of an urban scene, calculating probabilistic visibility for a given object from a specific view cell in the scene, has been presented by [14]. Nadler concept's extends the traditional deterministic visibility method. Nevertheless, the buildings are modeled as cylinders, and the main challenges of spatial analysis and building model were not tackled. Other methods were developed, subject to computer graphics and vision fields, dealing with exact visibility in 3D scenes, without considering environmental constraints. [7] used the aspect graph – a graph with all the different views of an object. Due to the computational complexity, all of these methods are not applicable to a large scene with near real-time demands, such as, multi agent's trajectory planning for patrolling applications.

III. THE PROBLEM

A. Problem Definition

The definition of the problem is commonly set according to predefined strategy planning. Patrolling strategy deals with different situations and therefore yields different formulation. We demonstrate these differences using the following division:

Corresponding author:
Oren Gal (email: orengal@technion.ac.il)

1. Force Planning - with a given set of locations that should be protected, we have to determine the minimal number of agents (K) with patrolling strategy which meets all constraints. This becomes the common case, when there is no available agents compatible with the patrolling mission

2. Force Division - locations are situated in different areas in an urban environment. We need to decide how to allocate the agents to all the areas in such way that it will be possible to find a patrolling strategy that meets all constraints in all the areas.

In our case, we deal with a force division problem combining relative deadline and visibility clustering. Given a set of N locations and K different types of agents (which are available for patrol at a given moment), Our method focus on finding a patrolling strategy, where each route for an agent passes through a number of locations. Patrolling strategy aims to minimize cost function, which is based on 3D visible volumes and meets the relative deadline constraints.

B. Locations

As part of the problem, our urban environment contains number of assets that should be protected, named as our Locations. Each location is characterized by:

1. Coordination's - its actual location in the environment, generated from Visibility Clustering based on 3D visible volumes analysis, detailed later on.

2. Required Protection - different locations may have different protection needs. As will be discussed later, we refer to the type of agent required for protection.

3. Relative Deadline - defined as the maximum time that may pass between consecutive visits for optimal patrolling (for example, to reduce the probability of being attacked to beneath a certain threshold).

C. Agents

Our Agents are modeled as Unmanned Ground Vehicles (UGVs). The UGVs are generalize and differentiate into two types of agents:

1. Large - Agent with long-range patrolling capabilities, sensing and target engaging abilities.

2. Small – Agent with limited ranges in the aspects described above.

Each type of agent has different average speed, dynamic and kinematic constraints with its own hourly operational costs and perception capabilities related to visibility analysis. There are many other factors to be considered, but it these are beyond the scope of this paper.

D. Agent's Dynamic Model

The four-wheeled car system (UGV) discussed in this work has rear-wheel drive and front-wheel steering. It uses the Ackerman steering configuration common to the standard automobile. This configuration assumes that all the wheels turn around the same point (rotation center) which is colinear with the rear axle of the car. Only the front wheels are capable of turning and the back wheels must roll without slipping [5]. This vehicle model is the first nonholonomic system ever studied in robotics [5, 6]. In the world of autonomous vehicles, it is not so prevalent, because there are many designs that provide greater mobility than the standard four-wheeled car.

Some other, simpler versions of the four-wheeled car have been used for experimentation in path planning. The Reeds-Shepp car [5] severely limits velocity choices by bounding velocity to v =1. It is only capable of maximum forward or maximum reverse speeds. The Dubins vehicle [5,8] limits velocity even further, to v=1, thus only allowing maximum forward speed. Figure 1 shows the model configuration [6]. L is the length of the car between the front and rear axles.

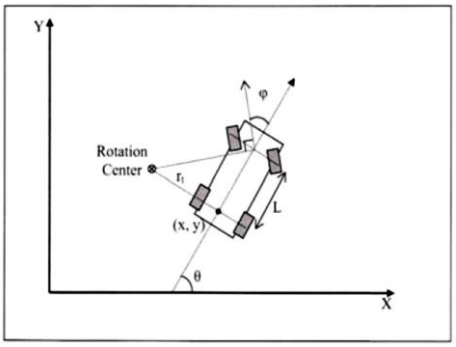


Figure 1 Four-Wheeled Car Model with Front-Wheel Steering [6].

The state vector is composed of two position variables (x,y) and an orientation variable (θ). The x-y position of the agent is measured at the center point of the rear axle. The control vector consists of the vehicle's velocity (v) and the angle of the front wheels (ϕ) with respect to the car's direction. The results in the kinematic equations of motion for the UGV are:

$$\dot{\underline{x}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v}{L}\tan(\varphi) \end{bmatrix}$$

Friction with the terrain is modeled by Bekker combined with Janosi-Hanomoto soil thrust models. We use the same model for large and small agents, where L value is set by agent's type.

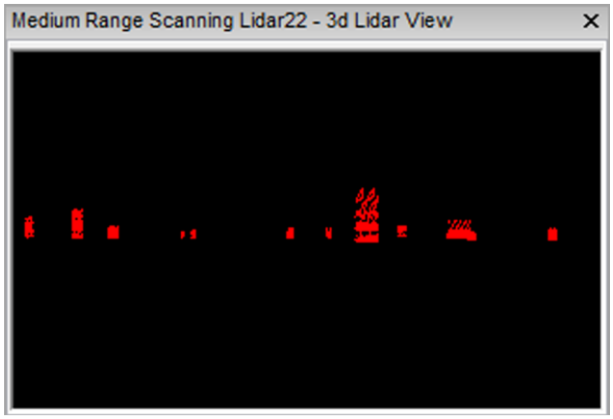
E. Agent's Perception Model

Our perception models are based on physical models simulated by ANVEL simulation environment [16], as seen in Figure 2 We use a LIDAR sensor for a large agent and cameras for a small ones.

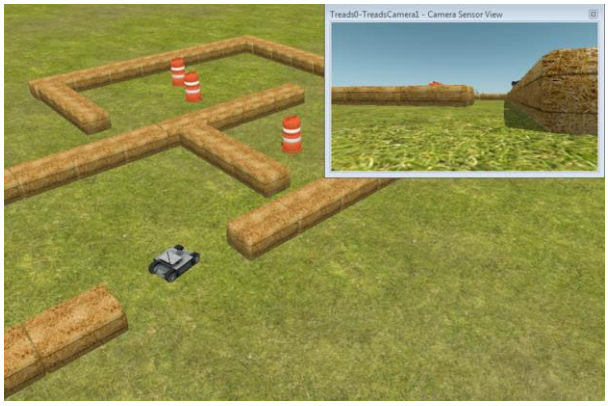
Large Agent Perception Model - ANVEL's parameterized LIDAR model uses a physics-based approach using raycasts. Particle effects, noise, and other errors are not included in this model.

Small Agent Perception Model - The ANVEL camera sensor can simulate a wide range of cameras, including sensors

with multiple lenses. Camera images can be analyzed to determine speed, detect obstacles, or recognize landmark features. Output from a camera is a series of images, captured at the rate specified in the sensor configuration.



(a)



(b)

Figure 2 (a) LIDAR sensor; (b) Camera Sensor window on top of the World View Pane in an ANVEL simualtor.

F. Route

As presented above, we follow the approach. In order to compare the cost of different strategies we need to define the term Route. Route is defined as a single cycle between all the locations allocated for a single agent in a certain order. In general, the cost of a route is calculated as the cost of the cycle, adding cost back and forth from the first location in the cycle to a predetermined port of origin. For every agent there will be one route in the patrolling strategy.

G. Patrolling Strategy

The collection of routes for all agents (one route per agent) which cover all the locations by one agent or another, is defined as Patrol Strategy. The total patrol cost is defined as the sum of all the routes' cost, where route cost includes a 3D visible volumes aspect integrated into visibility clusters. In section 4, we demonstrate how the best strategy (cost wise) is been calculated.

H. Main Assumptions

We have made the following assumptions:

1. **Agent Type** - Each location requires a different type of surveillance, and each type of agent has unique and different capabilities. Therefore, we assume that each location can be protected by only one type of agent: large or small. This assumption split our problem into two independent problems, each with its relevant locations and relative deadlines.

2. **Disjointed routes for each agent** - In order to simplify our problem, we assume that each agent is allocated to patrol a specific subset of locations. The subsets are disjointed sets, which create disjointed routes for each agent. The union of those subsets is the initial set of locations.

3. **Single visits to locations** - Another simplification of the problem is the assumption that along the route each location can be visited only once, and the problem can be described as TSP private case. Due to this assumption, when we have locations with short relative deadlines, there might be cases where no solution will be found. This will be addressed in the review of the solution technique.

4. **Patrolling along a cyclic path** - We assume that each patrol route is a cyclic path. The definition of cyclic path is a path which starts and ends at the same location .

5. **Traveling time between locations** - Shortest traveling time - the given traveling times are the shortest possible times between two given locations, based on the agent's dynamic model.

IV. Problem Formulation

As stated in the assumptions, we divide our problem into two identical independent problems. The formulation is similar for both types of agents.

A. Inputs

The inputs to this problem can be represented as an Undirected complete graph

$G = (V, E, t, g)$

- V - The vertices' set. Represents the locations in the problem generated by clustering visibility analysis, $|V| = N$.
- - The arcs set. Represents a route between two locations.
- $t : V \rightarrow \square^{+}$ - The relative deadline for each vertex.
- $g : E \rightarrow \square^{+}$ - The traveling time for each arc.
- C - The operational cost for each time unit.
- K - The number of agents.

B. Decision Variables

$$\Pi = \{\pi_1, \pi_2, ..., \pi_k\}$$

Denotes multi-agent patrol strategy composed of K single agents patrol strategy, where $\pi_k(j)$ is the j^{th} location visited by agent k on its patrol route. $\pi_k(0)$ is the location of the port of origin which is similar for all agents.

C. Objective Function & Constraints

Our problem formulation is based on the formulation given in (2).

$$\min_{\Pi} \left(\sum_{k=1}^{K} \sum_{j=1}^{J_k - 1} C \cdot g\left(\pi_k(j), \pi_k(j+1)\right) + \sum_{k=1}^{K} 2 \cdot C \cdot g\left(\pi_k(0), \pi_k(1)\right) \right)$$

s.t.

- $\sum_{k=1}^{K} I_k(i) = 1$
- $\forall i = 1, \ldots, N$
- $a_i = \sum_{j=1}^{J_k} I_{\pi_k(j)}(i) \geq 1$
-
- $\pi_k(1) = \pi_k(J_k)$
- $\forall k = 1, \ldots, K$
- $\sum_{j=A_{k_i}(l)}^{A_{k_i}(l+1)-1} g\left(\pi_k(j), \pi_k(j+1)\right) \leq t_i$
- $l = 1, 2, \ldots, a_i - 1, \quad \forall k = 1, \ldots, K, \quad \forall i \in \pi_k$
- $\sum_{j=1}^{A_{k_i}(1)-1} g\left(\pi_k(j), \pi_k(j+1)\right) + \sum_{j=A_{k_i}(a_i)}^{J_k - 1} g\left(\pi_k(j), \pi_k(j+1)\right) \leq t_i$
- $\forall k = 1, \ldots, K, \quad \forall i = 1, \ldots, N$

Where

1. $I_k(i) = \begin{cases} 1, i \in \pi_k \\ 0, else \end{cases}$

$I_k(i)$ is an indicator which gets the value 1 if location i belongs to patrol path π_k.

2. $I_{\pi_k(j)}(i) = \begin{cases} 1, i \in \pi_k(j) \\ 0, else \end{cases}$

$I_k(i)$ is an indicator which gets the value 1 if location i is the j^{th} visit in π_k.

3. J_k - The total number of locations which agent k is visits on its patrol.
4. a_i - The number of visits in location i.
5. A_{k_i} - Vector which contains the indices of the visits in location i in π_k. Therefore, $A_{k_i}(l)$ is the l^{th} index in vector A_{k_i}. For example, assume $\pi_k = (2,4,1,3,4,2)$, then $A_{k_4} = (2,5)$ since the indices of visits in location 4 in the vector π_k are 2 and 5. In addition $A_{k_4}(2) = 5$.

D. Formulation

Objective function - For every agent we calculate the cost of the route: total the traveling time from the first location on the route to the last one, multiply by cost, adding the cost of traveling back and forth to the point of origin.

Constraints -

- Disjointed routs for each agent.
- All locations are visited at least once.
- Each route begins and ends at the same location.
- All the relative deadlines are met.

E. Formulation Simplification

The formulation proposed above represents the general case of our problem and allows repeated visits on the patrol route. We simplify the formulation by referring to the private case in which each location will be visited exactly once.

$$\min_{\Pi} \left(\sum_{k=1}^{K} \sum_{j=1}^{J_k - 1} C \cdot g\left(\pi_k(j), \pi_k(j+1)\right) + \sum_{k=1}^{K} 2 \cdot C \cdot g\left(\pi_k(0), \pi_k(1)\right) \right)$$

- $\sum_{k=1}^{K} I_k(i) = 1$
- $\forall i = 1, \ldots, N$
- $\sum_{j=1}^{J_k} I_{\pi_k(j)}(i) = 1$
- $\forall k = 1, \ldots, K, \forall i \in \pi_k$
- $\pi_k(1) = \pi_k(J_k)$
- $\forall k = 1, \ldots, K$
- $\sum_{j=1}^{j=J_k} g\left(\pi_k(j), \pi_k(j+1)\right) \leq t_k$
- $\forall k = 1, \ldots, K$
- $t_k = \min_{\forall i \in \pi_k}(t_i)$

The main differences between the formulations are: in constraint (2) we make sure each location is visited only once in a given agent's route, and in constraint (4) we make sure all relative deadlines are met. The total time required for a certain agent to travel between all its locations and back to the first location is shorter than the minimal deadline of the locations in its path.

F. Upper bound to the problem

The upper bound to our problem, given the assumptions above, is:

$$K \cdot \left(C \cdot \max_{i=1 \ldots N}(t_i) + 2 \cdot C \cdot g\left(orgloc, maxloc\right) \right)$$

where

1. $orgloc$ - The location of the port of origin.
2. $maxloc$ - The location furthest from the port of origin.

This is the maximal relative deadline of all the locations in each agent's route, with the addition of the traveling time back and forth from the location furthest from the port of origin for each of the K agents.

V. Patrolling Strategy Algorithm

As mentioned in the Introduction section, we followed the approach, which means that each agent has a unique subset of locations to patrol in. These locations are clustered based on 3D visible analysis. Our goal is to find a patrol strategy, which is the division into clusters and the route within each cluster, which on one hand yields minimum cost and on the other hand allows the most visible route. Our patrolling strategy algorithm consists of five stages, detailed below.

A. Dividing the locations into K spatial clusters

The N locations will be partitioned into K disjointed subsets. Each agent k will have a unique subset of locations that will be visited by a specific agent. Since we are dealing with patrolling strategies and a cost that is based on the distances traveled, we assumed a division into clusters based on 3D visibility analysis [21].

The method is as follows: We apply spatial analysis clustering method that will be used as cluster centers points and which then assigns the remaining locations to their nearest cluster center (assigns to one out of K centers). The result is K disjointed clusters respective to K agents, and we denote the number of locations in each cluster by n_k.

Spatial Clustering Analysis

Our data set $\{X_{ij}\}$, i=1,2,..,n, j=1,2,..,p, consists of p features measured on n independent observation, where observation points marked with blue circles are illustrated in Figure 3. Our data clustered the data into k clusters, $C_1, C_2, .. C_k$. For cluster r, denoted as C_r with n_r observations:

$$V_r = \sum_{i \in C_r} \sum_{j \in C_r} \left\| V(x_i) - V(x_j) \right\|$$
$$T_k = \sum_{r=1}^{k} \frac{1}{S} V_r \quad \textbf{(1)}$$
$$V_r = \sum_{i \in C_r} \left\| V(x_i) - V(\bar{x}) \right\|$$

where V(x) denotes the visible volumes from a viewpoint x bounded inside the total volume S, V_r is the sum of the absolute visibility differences of all viewpoints from their cluster visibility mean, and the normalized visible volumes T_k for all clusters r=1..k, is called dispersion.

Similarly to many other methods estimating the number of clusters [15; 22], our concept is based on define reference data sets distributed uniformly inside bounding volume S. We define our reference data sets with the same size of the original data set X, and calculate the dispersion of theswrge datasets, T_k^*.

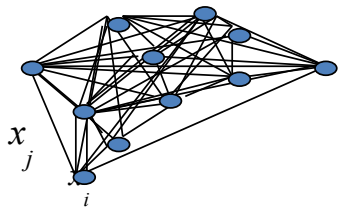


Figure 3 Pedestrians' location architecture based on monitoring datasets, observation

Based on F statistic, datasets are analyzed, where adding another cluster does not give a better modeling of the data, also known as F-test criteria. By setting a group's visibility variance, the number of clusters can be estimated efficiently:

$$SVC_n(k) = T_k^* - T_k \quad \textbf{(2)}$$

Fast and efficient visibility volume computation from a specific viewpoint, bounded in volume S, is presented in the next subsection.

SVC steps can be summarized as follows:

1. Calculate the sum of absolute visibility differences of all points from their cluster visibility mean. Normalize this sum for all possible clusters T_k, also called dispersion.
2. Generate a set of reference datasets, simulated by a uniform distribution model inside bounding volume S.
3. Calculate the dispersion of each of these reference datasets, and calculate their mean visibility values.
4. Define SVC for each possible number of clusters as: Expected dispersion of reference datasets - Dispersion of original dataset.

Originally, F statistic was used to test the significance of the reduction in the sum of squares as we increase the number of clusters [19]. From a certain k, dividing a dataset into k+1 clusters decreases the value of the F-test function which depends on k.

Approximated F-test function: Assuming that T_k is the partition of n instances into k clusters, and T_{k+1} is obtained by T_k splitting one of the clusters, then the overall mean ratio can be approximated as:

$$F_k = \frac{SVC_n}{SVC_{n+1}} \quad \textbf{(3)}$$

We adapted aspects of previous F statistic theory for visibility analysis. More detailed F statistical analysis can be found in [19]. The spatial meaning of this mathematical clustering formulation can be simplified as a group of viewpoints with minimal difference to the average visible volume in the same bounding box.

B. Set feasible patrol strategy

Given a certain partition into spatial clusters, we have to determine a patrol strategy for each cluster generated from 3D visible volumes analysis. The meaning of patrol strategy is an order of locations that will be visited by one of the agents. After investigating different methods to create these patrol strategies, we decided to apply the TSP technique due to its simplicity and efficiency.

When dealing with cases with a small dimension problem, a TSP solution can be calculated. The classic TSP problem deals with a list of cities and the distances between each pair of cities, where the question is: what is the shortest possible route that visits each city exactly once and returns to the city of origin? By applying the TSP technique in each cluster, we create a patrol strategy that visits each location exactly once and returns to the origin location. It is important to emphasize that there may be a division into clusters in which our suggested patrol strategy with the lowest cost may not be feasible. One of the TSP restrictions. Of visiting every location exactly once, may not enable us to meet the relative deadline constraints. Later on, a case with such an infeasible solution is considered.

In order to calculate the TSP solution we implement the following steps: First, setting K spatial clustering, and in each cluster calculating the distance from each location to the origin in order to choose the closest location as the first station on the route. The algorithm checks all the permutations of the locations' order in terms of traveling time (sum up the traveling time of the given order). The shortest path is the one with no cutting arcs (due to Triangle Inequality).

C. Feasibility of the patrol strategy

Given a patrol strategy in a cluster, feasibility should be checked. Feasible patrol strategy is defined as a patrol which meets the relative deadlines of all the locations in the cluster. If traveling time between the locations (according to the chosen permutation) is less than the minimum relative deadline of those locations, it means that all the relative deadlines were met and the patrol is a feasible one.

Total solution (consist of K patrol strategies) is defined as a feasible one, based on the feasibility of each of the K patrol strategies separately.

D. Fine Calculation

In some cases, there is no feasible solution. Since we have limited our solution to paths with no repeated visits, the TSP technique might not always find a feasible solution. For every total solution which is non-feasible (one or more of the K patrol strategies are non-feasible), we calculate a *fine*. The fine in a cluster is proportional (multiplying by a constant) to the following: [The traveling time between the locations according to the chosen permutation]-[The minimum relative deadlines of the locations in the cluster]. We will denote this difference as a deviation from deadline, and the total deviation is the sum of the deviations of K clusters. The total fine of a solution is the total deviation multiplying by a constant (constant F).

E. Finding the optimal solution and calculating total cost

For each partition into K spatial clusters, the optimal patrol strategy in each cluster is the one that yields minimum traveling time between the locations. In order to find the total traveling time of a specific patrol strategy, we add the time it takes to arrive to and from the first location from the origin to the traveling time between the locations. The total time of a solution will be the sum of the total traveling time of K patrol strategies.

The optimal solution under our assumptions (see section 2), can be calculated as follows:

- If there are one or more feasible solutions we will chose the patrol strategy that yields the minimal traveling time, calculated as described above. The total cost of the solution will be by multiplying the total traveling time of the solution in a constant that represents the cost of one traveling hour (constant C).

- If there is no feasible solution in any of the partitions, the patrol strategy is chosen which the minimal sum of fines from all K clusters. The total cost of this kind of solution is the cost of the total traveling time (the same as for feasible solutions) plus the total fine.

VI. Algorithm Implementation

We simulate and demonstrate our concept using a virtual simulation environment called ANVEL (Autonomous Navigation and Virtual Environment Laboratory). ANVEL is a virtual simulation tool that specializes in unmanned ground vehicle (UGV) technologies and applications. ANVEL provides a modular virtual environment for developing and testing UGV technologies in an interactive, visual manner. The application simulates a wide collection of variables that are involved in vehicle operation, from physical mobility, to detecting and analyzing the world through sensors, to the controlling logic that guides vehicles through various missions or tasks. ANVEL allows us to build virtual test environments, manipulate parameters of key subsystems, change aspects of the terrain, add or remove sensors, and interactively test vehicles and robots under a variety of conditions. ANVEL is also highly customizable. It combines realistic graphics with sophisticated algorithms to create a stunning simulation experience that rivals real-world vehicle testing.

A. Program Inputs

The inputs of the program are as follows:

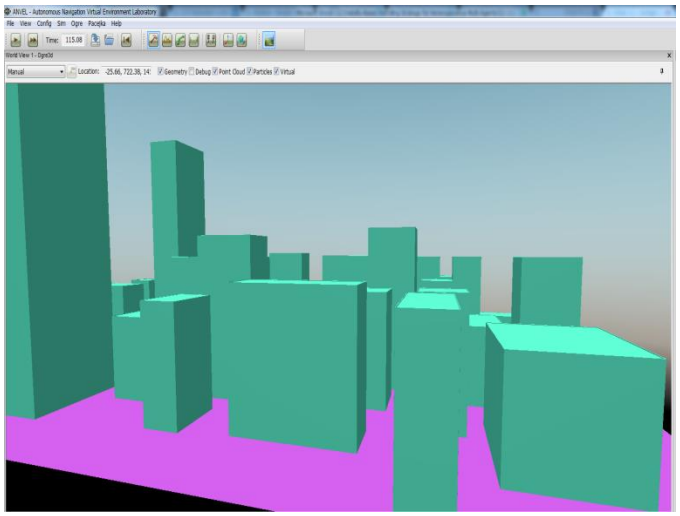
- N- number of locations
- K- number of agents
- Size of patrolling area
- Agent's average speed and physical dimensions
- C- The cost per one operational hour
- F- The cost per one hour of deviation from the deadlines (the fine cost)

We assumed 10 locations in total ($N=10$) and 3 agents in total ($K=3$), C= $\$100$, F= $\$1000$ and the average speed of the agent is 65 km/h. We randomly generated the relative deadline of each location (9h on average) and also the coordinates of the N locations in the patrolling area (500 km^2).

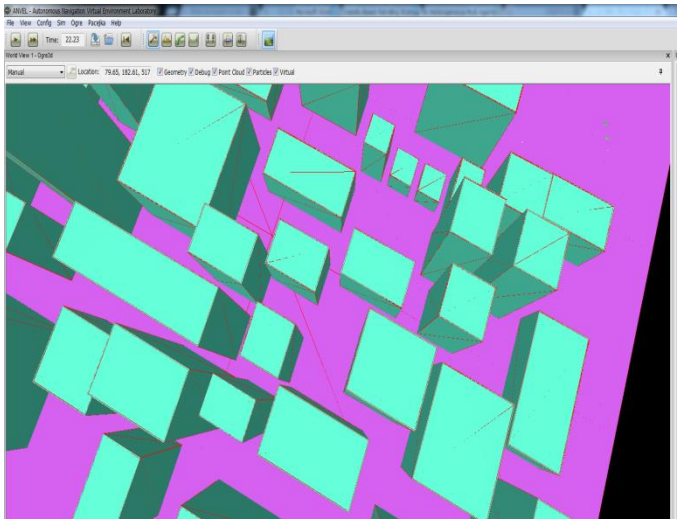
B. Problem Dimensions

The problem has the following dimensions: ten locations for small agents, six locations for large agents, and three small agents patrol. Our tested urban environment can be seen in

Figure 4. We set two large agents and one small agent on patrol at a given time. Agents in a simulated environment can be seen in Figure 5. Perception capabilities of each agent can be seen in Figure 6 and Figure 7. Locations are changed according to the initial position of the agents at each simulation. Locations are set as the outcome of our visibility clustering analysis, as seen in Figures 8 and 10.



(a)



(b)

Figure 4 Our test bed urban environment model in ANVEL, (a) Side view; (b) Top view.

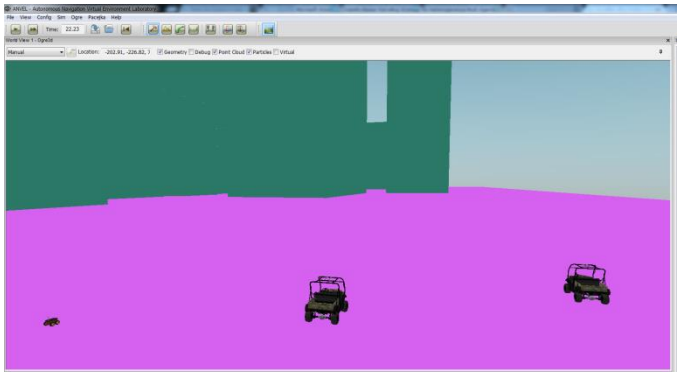


Figure 5 Three agents in our simulation, two large agents on the right side and a small agent on the left.

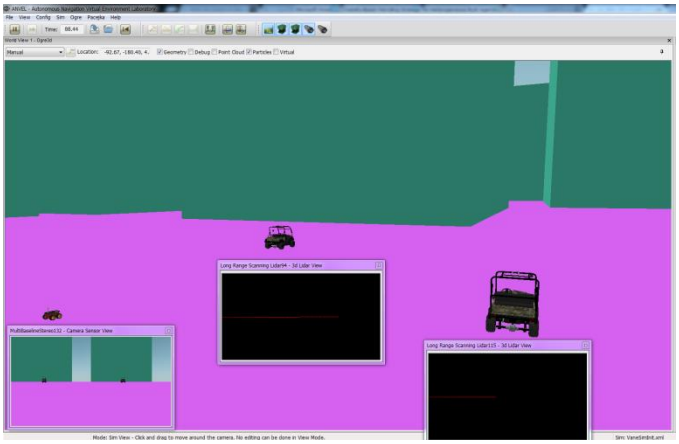


Figure 6 Perception sensor; LIDAR sensor for large agents and camera sensor for a small agent.

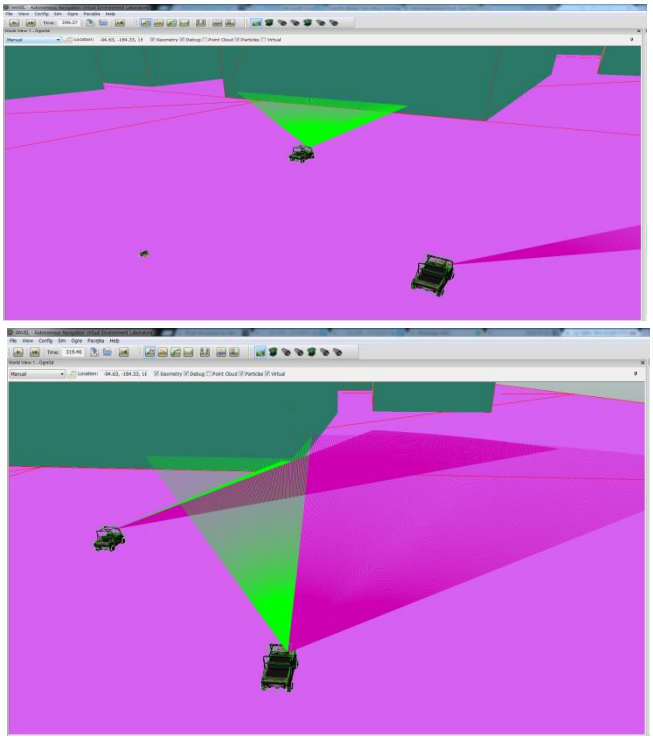


Figure 7 LIDAR perception capabilities during time.

C. Simulations Results

Based on spatial clustering and patrolling strategy, simulation results are presented in this section.

In the first case, coordinates of the 10 locations presented in Figure 8:

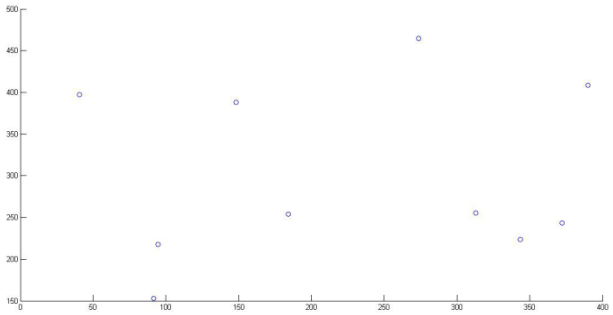


Figure 8 Location coordinates based on Visibility Clustering.

The patrolling graph can be seen in Figure 9:

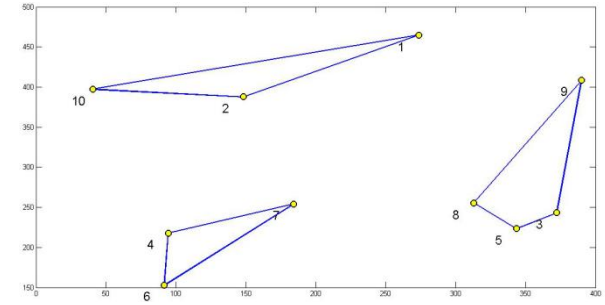


Figure 9 Patrolling Strategy Trajectories According to Problem Constraints.

Patrol #1 [10, 1, 2, 10], Patrol #2 [6, 4, 7, 6] and Patrol #3 [8, 5, 3, 9, 8]. Total time of the solution is (h) 48.885, and total deviation from deadlines is (h) 0. The total cost of the solution is 4888.5. The total time of the solution is the sum of the total traveling time of all three patrols. For example, the total traveling time of patrol #3 is the time to and from location 8 plus the traveling time between the locations in the cluster, meaning the traveling time from location 8 to 5, 5 to 3, 3 to 9 and 9 to 8.

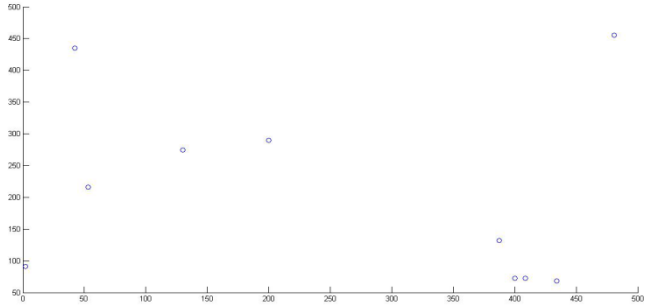


Figure 10 Location Coordinates based on Visibility Clustering in Second Simulation.

The total deviation from deadlines is zero, i.e. the solution is feasible and there is no need to pay a fine.

The total cost of the solution is $100*48.885=4888.5$.

Our second simulation demonstrates patrol strategies that are non-feasible. Coordinates of the 10 locations are presented in Figure 10.

The patrolling graph can be seen in Figure 11

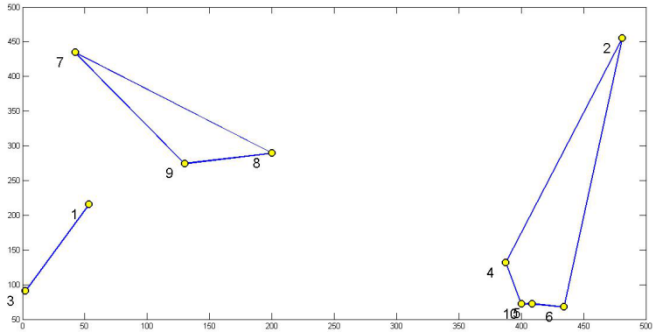


Figure 11 : Patrolling Strategy Trajectories According to Problem Constraints.

The program prints the following results: Patrol #1 [3, 1, 3], Patrol #2 [10, 4, 2, 6, 5, 10], Patrol #3 [9, 7, 8, 9], as seen in Figure 11. Total time of the solution is (h) 48.65, total deviation from deadlines is (h)4.53 and the total cost of the solution is ($) 9395. The total deviation from deadlines is the sum of the deviations of the patrol strategies that are non-feasible (it is not necessary for all 3 patrol strategies to have a deviation). The total cost of the solution is $100*48.65+1000*4.53=9395\$$

Another interesting example is of a case in which one of the patrol strategies contains only one location. The reason for this is the coordinates of this location, which are far from all the other locations.

In the first case, coordinates of the 10 locations are presented in Figure 12:

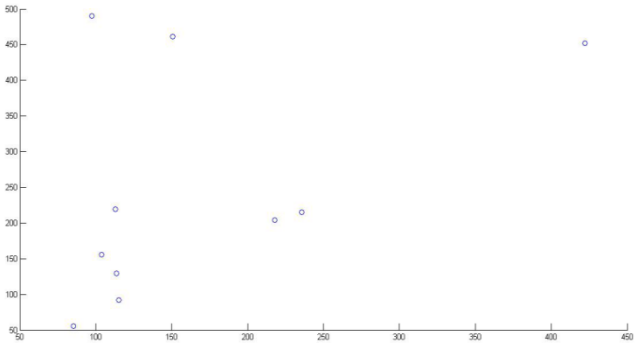


Figure 12 Location coordinates based on Visibility Clustering in Third Simulation.

The program prints the following results: Patrol #1 [8, 9, 1, 7, 3, 10, 4, 8], Patrol #2 [2, 6, 2], Patrol #3 [5], as seen in Figure 13. Total time of the solution is (h) 46.88, total deviation from deadlines is (h) 0, and the total cost of the solution is ($) 4688.

VII. Discussion

Our research tackled a simplified version (given our assumptions) of an $NP-Hard$ problem, as described in the introduction. In this section we would like to discuss other complicated cases which are received by eliminating one or more of our assumptions, and cases which are received by different problem definitions. We suggest a general solution

approach for these cases and conclude with recommendations for possible research extensions.

1. **Finding a patrol strategy when repeated visits are allowed** - as we described in the second step of our solution technique (detailed in section (4)), the patrol strategy in each cluster will be found by using TSP technique, which does not allow repeated visits apart from the first location in the path. Naturally, there might be a path with repeated visits which is a feasible solution (all the relative deadlines are met), but the TSP technique will "miss" this solution. Using the TSP technique is one of the simplifications that we have made, and we are aware of the fact that we might miss feasible solutions. In order to deal with a case in which there are no feasible solutions at all, we added the fine calculation, but here we would like to present, in short, a heuristic approach for finding a patrol strategy with repeated visits.

This approach is based on [2] and the idea is as follows: For each cluster, the patrol strategy receives a feasible solution π_k (a patrol route with J_k visited locations, $J_k > n_k$ since repeated visits are allowed and the patrol must visit every location in the cluster at least once, with $\pi_k(1) = \pi_k(J_k)$). Given a starting location (which is also the end location), the method uses the following algorithms:

- Forward Checking - given the last location in the patrol forming strategy $\pi_k(j-1)$, this algorithm finds all the possible locations for $\pi_k(j)$ that will satisfy the second and third constraints. All those locations are added to a vector F_j.

- Recursive call - given the last location in the patrol forming strategy $\pi_k(j-1)$ and the index of the following step j, it checks if the patrol has not ended ($\pi_k(1) = \pi_k(j-1)$) with all constraints satisfied. If that is not the case, it calls *Forward checking* and run itself recursively on every location in F_j and the index $j+1$ until a patrol strategy is achieved. Meaning, we have a route which is a closed path and every location is visited at least once. When $F_j = \varnothing$ the algorithm backtracks till a different location can be chosen and continues that route.

2. **Non-disjointed routes** - one of the simplifications that was made is to assume that the route of each agent is disjointed from all other routes. The reason for that was to avoid collisions between the agents and to simplify our problem. Obviously, in reality, the routes do not have to be disjointed. When eliminating this assumption the complexity of the problem rises, due to the fact that there are many more cluster partition options. One can suggest a heuristic approach to try and find a solution to the problem. First, preform the same partition of the location into clusters (or a similar partition method based on geographical proximity) and then check if there is a solution with disjointed routes. If there is no such solution, find the locations for which the relative deadline constraint is not met. Then, try and add each of those locations to another cluster based on a proximity criterion, such as the shortest average distance to all locations in the cluster and so forth. Now these locations are visited by more than one agent, and therefore the time between consecutive visits is reduced to a point that may yield a feasible solution.

3. **Joint routes for a number of agents while the goal is to minimize the maximal idle time of all locations** - This is the approach mentioned in the introduction (see section (1)). In [1] a heuristic algorithm is proposed to solve this problem:

- Create an Outer-planner Graph from all given locations. This is done by connecting all the locations with arcs that do not cross one another. Allocate all agents to this graph and calculate the maximal idle time.

- By removing two arcs at a time from the graph we create two separate graphs. Remove all possible pairs of arcs and, for each graph pair created, find the agent allocation (meaning the number of agents allocated to each graph) that minimizes the maximal idle time of both graphs.

- Find the division that yields the minimal maximal idle time, and for each graph preform the same process as described in the previous stage.

- Continue until the maximal idle time cannot be reduced, or when the only possible allocation is a single agent to a graph.

4. **Locations that require protection from both type of agents** - in our solution, we assumed that each agent can be protected by only one type of agent. The reason for that was to separate the initial problem into two disjoin problem. That way, we can solve each problem separately. Obviously, in reality, there exist locations that require protection by both types of agents (we will refer to them as hybrid locations). As long as there are different relative deadlines for each type of agent, then we can continue referring to the problem as two separate problems. The complication begins when the relative deadline is unified, meaning the maximal time that may pass between consecutive visits of any type of agent. A possible way of approaching this case without unifying the problems (which may yield a large-dimension problem that is harder to solve) is as follows:

- Solve the separate problems.

- If no feasible solution exists for one of the problems, that is due only to a problem in meeting one or more hybrid locations' relative deadline constraints. Continue to the next stage

- Combine the solution with the total minimal deviation from the relative deadlines of the hybrid locations with feasible solutions (or those with minimal deviation as well) for the other type of agents and update the time between consecutive visits to hybrid locations accordingly. Check if the relative deadlines can now be met.

- If no feasible solution can be found that way, the fine method we introduced in this paper may be used.

VIII. Conclusion and Future Work

In this research, we presented visible trajectories planning for patrolling application using heterogeneous multi agents in 3D urban environments. Using the ANVEL simulation environment, spatial clustering method using visibility analysis was demonstrated. As part of our simulations two kinds of agents modeled with different kinematic and perception capabilities.

Patrolling strategy was formulated as Traveling Salesman Problem (TSP), with relative deadline UniPartition approaches based on visibility clusters.

Simulations result showed implementation on different cases, using large and small agents in urban environment scenarios where trajectory cannot be found occasionally. Certain problem formulations are discussed and solutions are suggested for further research, such as: finding a patrol strategy when repeated visits are allowed; non-disjointed routes as part of our patrol graph; joint routes for a number of agents while the goal is to minimize the maximal idle time of all locations, and other cases of locations that require protection from both types of agents.

ACKNOWLEDGEMENT

The authors would like to thank Nofar Winterman Ben Levav and Michal Penn from the Technion, for there help during this research.

REFERENCES

[1] N. Agmon, D. Urieli and P. Stone. Multiagent Patrol Generalized to Complex Environmental Conditions. Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence. San Francisco, California, 2011.

[2] N. Basilico, N. Gatti, and F. Amigoni. Developing a deterministic patrolling strategy for security agents. In proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology- Vol. 2, pages 565-573. IEEE computer Society, 2009.

[3] Y. Chevaleyre. 2004. Theoretical analysis of the multi-agent patrolling problem. In Proc. of IAT'04.

[4] R. Wendolsky, S. Scheuerer. A Cluster Based Scatter Search Heuristic for the Vehicle Routing Problem. University of Regensburg Discussion Papers in Economics No. 415, November 2006.

[5] Siegwart, R. and Nourbakhsh, I.R., Introduction to Autonomous Mobile Robots, The MIT Press, 2004.

[6] Lewis, L.R., Rapid Motion Planning and Autonomous Obstacle Avoidance for Unmanned Vehicles, Master's Thesis, Naval Postgraduate School, Monterey, CA, December 2006

[7] H. Plantinga, and R. Dyer, Visibility, Occlusion, and Aspect Graph, The International Journal of Computer Vision, 5,137-160, (1990)

[8] Y. Doytsher, and B. Shmutter, Digital Elevation Model of Dead Ground, Symposium on Mapping and Geographic Information Systems (Commission IV of the International Society for Photogrammetry and Remote Sensing), Athens, Georgia, USA, (1994)

[9] F. Durand, 3D Visibility: Analytical Study and Applications, PhD thesis, Universite Joseph Fourier, Grenoble, France, 1999

[10] J. Wang, G.J. Robinson, and K. White, A Fast Solution to Local Viewshed Computation Using Grid-based Digital Elevation Models, Photogrammetric Engineering & Remote Sensing, 62, 1157-1164, (1996)

[11] J. Wang, G.J. Robinson, and K. White, Generating Viewsheds without Using Sightlines, Photogrammetric Engineering & Remote Sensing, 66, 87-90, (2000)

[12] C. Ratti, The Lineage of Line: Space Syntax Parameters from the Analysis of Urban DEMs', Environment and Planning B: Planning and Design, 32,547-566, (2005)

[13] L. De Floriani, and P. Magillo, Visibility Algorithms on Triangulated Terrain Models, International Journal of Geographic Information Systems, 8, 13-41, (1994)

[14] B. Nadler, G. Fibich, S. Lev-Yehudi, and D. Cohen-Or, A Qualitative and Quantitative Visibility Analysis in Urban Scenes, Computers & Graphics, 5, 655-666, (1999)

[15] A. Gordon, Classification (2nd ed.), London: Chapman and Hall/CRC Press, 1999.

[16] ANVEL: http://anvelsim.com/

[17] Brumitt, B. L., Stentz, A.: Dynamic Mission Planning for Multiple Mobile Robots. Proceedings 1996 IEEE International Conference on Robotics Automation , 4, 2396-2401, (1996)

[18] Brumitt, B. L., Stentz, A: GRAMMPS: A Generalized Mission Planner for Multiple Mobile Robots In Unstructured Environments. Proceedings 1998 IEEE International Conference on Robotics Automation , 2, 1564-1571, (1998)

[19] Guo, Y., Lynee, P. E. :A Distributed and Optimal Motion Planning Approach for Multiple Mobile Robots. Proceedings of 2002 IEEE International Conference on Robotics Automation Washington, DC , 3, 2612-2619, (2002).

[20] Lumelsky, V. J., Harinarayan, K. R. :Decentralized Motion Planning for Multiple Mobile Robots: The Cocktail Party Model. Autonomous Robots , 4 (1), 121-135, (1997).

[21] O.Gal, Y. Doytsher. "Spatial Visibility Clustering Analysis in Urban Environments Based on Pedestrians' Mobility Datasets", Int. Conf. of GeoProcessing 2014, pp. 38-44, Spain.

[22] T. Schelhorn, D. Sullivan, M. Haklay, "STREETS: An agent-based pedestrian model," http://www.casa.ucl.ac.uk/streets.pdf. , 1999, [accessed February 2014].