Paper

Military route planning in battlefield simulation: effectiveness problems and potential solutions

Zbigniew Tarapata

Abstract — Path searching is challenging problem in many domains such as simulation war games, robotics, military mission planning, computer generated forces (CGF), etc. Effectiveness problems in military route planning are related both with terrain modelling and path planning algorithms. These problems may be considered from the point of view of many criterions. It seems that two criterions are the most important: quality of terrain reflection in the terrain model and computational complexity of the on(off)-line path planning algorithm. The paper deals with two above indicated problems of route planning effectiveness. Comparison of approaches used in route planning is presented. The hybrid, terrain merging-based and partial path planning, approach for route planning in dynamically changed environment during simulation is described. It significantly increase effectiveness of route planning process. The computational complexity of the method is given and some discussion for using the method in the battlefield simulation is conducted. In order to estimate how many times faster we can compute problem for finding shortest path in network with n big squares (b-nodes) with relation to problem for finding shortest path in the network with V small squares (s-nodes) acceleration function is defined and optimized.

Keywords — battlefield simulation, route planning, shortest paths, effectiveness problems, computational complexity.

1. Introduction

For many years in military applications a simulated battlefield is used for training military personnel. There are at least three ways to provide the simulated opponent:

- two groups of trainees in simulators may oppose each other (often used);
- human instructors who are trained to behave in a way that mimics the desired enemy doctrine (seldom used);
- computer system that generates and controls multiple simulation entities using software and possibly a human operator.

The last approach is known as a semi-automated force (SAF or SAFOR) or a computer generated force (CGF). CGF is used in military distributed interactive simulation (DIS) systems to control large numbers of autonomous battlefield entities using computer equipment and software rather than humans in simulators.
The advantages of CGF are well-known [17]:

1) they lower the cost of a DIS system by reducing the number of standard simulators that must be purchased and maintained;
2) CGF can be programmed, in theory, to behave according to the tactical doctrine of any desired opposing force, and so eliminate the need to train and retrain human operators to behave like the current enemy;
3) CGF can be easier to control by a single person than an opposing force made up of many human operators and it may give the training instructor greater control over the training experience.

As an inseparable part of CGF, modules for route planning based on the real-terrain models are used. For example in modular semi-automated forces (ModSAF) in module "SAFsim", which simulates the entities, units, and environmental processes the route planning component is located [14]. Moreover, automated route planning will be a key element of almost any automated terrain analysis system that is a component of a military command and control system. In the work [1] authors describe a combined on-road/off-road planning system that was closely integrated with a geographic information system and a simulation system. Routes can be planned for either single columns or multiple columns. For multiple columns, the planner keeps track of the temporal location of each column and insures they will not occupy the same space at the same time. In the same paper the hierarchic route planner as integrate part of predictive intelligence military tactical analysis system (PIMTAS) is discussed. In the paper [8] authors presented an on-going efforts to develop a prototype for ground operations planning, the route planning uncertainty manager (RPLUM) tool kit. They are applying uncertainty management to terrain analysis and route planning since this activity supports the commander's scheme of maneuver from the highest command level down to the level of each combat vehicle in every subordinate command. They extend the PIMTAS [1] route planning

software to accomodate results of reasoning about multiple categories of uncertainty. Authors of the paper [3] presented route planning in the close combat tactical trainer (CCTT).

Kreitzberg [11] has developed the tactical movement analyzer (TMA). The system uses a combination of digitized maps, satellite images, vehicle type and weather data to compute the traversal time across a grid cell. TMA can compute optimum paths that combine both on-road and off-road mobility, and with weather conditions used to modify the grid cost factors. The smallest grid size used is approximately 0.5 km. Author uses the concept of a signal propagating from the starting point and uses the traversal time at each cell in the array to determine the time at which the signal arrives at neighboring cells. Other researchers have chosen to decompose the map into regions that are defined by having a constant traversability across the region [1, 9, 16, 19, 20, 27]. The advantage of this approach is that the number of regions will, in general, be far fewer than the number of grid cells. The disadvantages include difficulty in defining the center of the region and the computation difficulties in determining the optimum paths between two adjacent cells. The optimum region-to-region path can be obtained by using either Dijkstra's continuous algorithm (DCA) developed by Mitchell [15]. In many cases, a multiresolution simulation modelling is used to simplify complex battlefield processes [4, 6, 16, 17].

As integrated part of route planning modules the terrain database-based model is being used. Terrain data can be as simple as an array of elevations (which provides only a limited means to estimate mobility) or as a complex as an elevation array combined with digital map overlays of slope, soil, vegetation, drainage, obstacles, transportation (roads, etc.) and the quantity of recent weather. For example in [1] authors describe heterogeneous reasoning and mediator environment system (HERMES) will allow the answering of queries that require the interrogation of multiple databases in order to determine the start and destination parameters for the route planner.

There are a few approaches in which the map (representing a terrain area) is decomposed into a graph [1, 9, 19, 20]. All of them first convert the map into regions of *go* (open) and *no-go* (closed). The *no-go* areas may be considered as obstacles and are represented as polygons. A few ways for consider the map can be used, for example: visibility diagram, Voronoi diagram, straight-line dual of the Voronoi diagram, edge-dual graph, line-thinned skeleton, regular grid of squares, grid of homogeneous squares coded in quadtree system, etc.

Effectiveness problems in military route planning are related both with terrain modelling and path planning algorithms. These problems may be considered from the point of view of many criterions. It seems that two criterions are the most important: quality of terrain reflection in the terrain model (visibility diagram, Voronoi diagram, regular grid of terrain squares, etc.) and computational complexity of the on(off)-line path planning algorithm. The paper deals with above indicated problems of route planning effectiveness.

In the next section we will discuss in details route planning approaches.

2. Comparison approaches used in route planning

It was said in the previous section that we will deal with effectiveness of two problems of battlefield simulation:

- terrain reflection in the terrain model used in battlefield simulation;
- military route planning using one of terrain models.

If terrain models are concerned a few ways for considering the map were listed in the previous section: Voronoi diagram, straight-line dual of the Voronoi diagram (the Delaunay triangulation), visibility diagram, edge-dual graph, line-thinned skeleton, regular grid of squares, grid of homogeneous squares coded in quadtree system.

The polygonal representations of the terrain are often created in database generated systems (DBGS) through a combination of automated and manual processes [19]. It is important to say that these processes are computationally complicated but are conducted before simulation (during preparation process). Typically, an initial polygonal representation is created from the digital terrain elevation data through the use of an automated triangulation algorithm, resulting in what is commonly referred to as a triangulated irregular network (TIN). A commonly used triangulation algorithm is the Delaunay triangulation. Definition of the Delaunay triangulation may be done via its direct relation to the Voronoi diagram of a set, S, of N 2D points: the straight-line dual of the Voronoi diagram is a triangulation of S.

The **Voronoi diagram** is the solution to the following problem: given a set S of N points in the plane, for each point p_i in S what is the locus of points (x, y) in the plane that are closer to p_i than to any other point of S?

The **straight-line dual** is defined as the graph embedded in the plane obtained by adding a straight-line segment between each pair of points of S whose Voronoi polygons share an edge. Figure 1 depicts an irregularly spaced set of points S, its Voronoi diagram, and its straight-line dual (i.e. its Delaunay triangulation).

The **edge-dual graph** is essentially an adjacency list representing the spatial structure of the map. To create this graph, we assign a node to the midpoint of each map edge which does not bound an obstacle (or the border). Special nodes are assigned to the start and goal points. In each non-obstacle region, we add arcs to connect all nodes at the midpoints of the edges which bound the same region. The fact that all regions are convex guarantees that all such arcs cannot intersect obstacles or other regions. Example of the edge-dual graph is presented in Fig. 2.

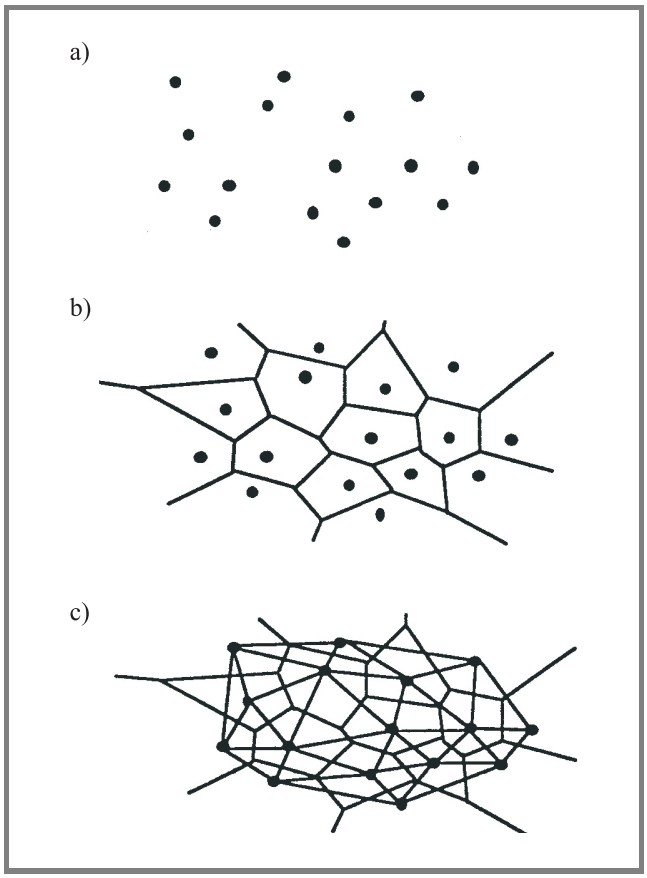

Fig. 1. Voronoi diagram and its Delaunay triangulation [19]: (a) a set S of N points in the plane; (b) the Voronoi diagram of S; (c) the straight-line dual of the Voronoi diagram (the Delaunay triangulation).

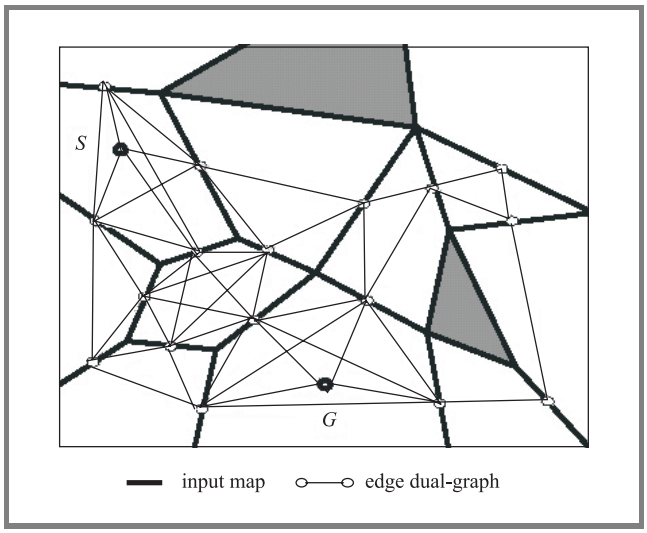

Fig. 2. Edge-dual graph. Obstacles are represented by filled polygons.

The **visibility graph**, is a graph whose nodes are the vertices of terrain polygons and whose edges joint pairs of nodes for which the corresponding segment lies inside polygon. An example is shown in Fig. 3.

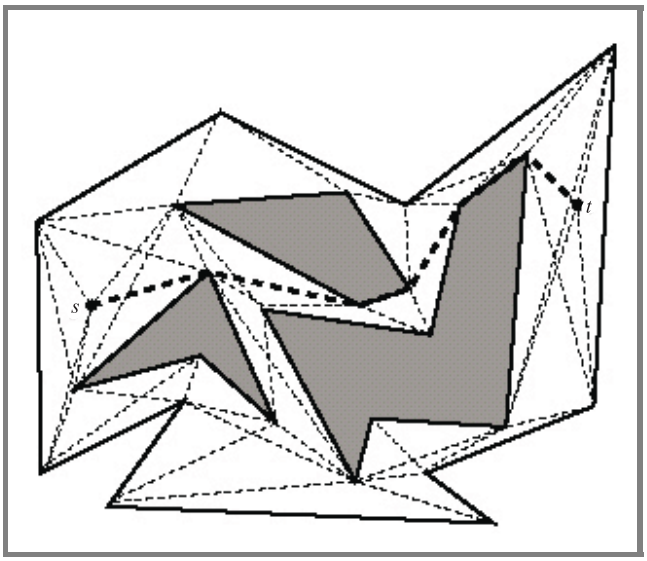

Fig. 3. Visibility graph [15]. There is marked shortest geometric path from source node s to destination t. Obstacles are represented by filled polygons.

The **regular grid of squares** divides terrain space on the squares with the same size and each square is treated as having homogeneity from the point of view of terrain characteristics. An example of this approach will present in the next sections (see Fig. 6 and Fig. 7).

The **grid of homogeneous squares coded in quadtree system** divides terrain space on the squares with heterogeneous size. The size of square results from its homogeneity according to terrain characteristics. Example of this approach was presented, e.g. in [29].

If paths planning approaches used in battlefield simulation are concerned, there are four main approaches [10]: free space analysis, vertex graph analysis, potential fields, grid based algorithms.

In the **free space approach**, only the space not blocked and occupied by obstacles is represented. For example, representing the center of movement corridors with Voronoi diagrams [19] is a free space approach (see Fig. 1).

Advantage of Voronoi diagrams is that they have efficient representation.

Disadvantages of Voronoi diagrams:

- they tend to generate unrealistic paths (paths derived from Voronoi diagrams follow the center of corridors while paths derived from visibility graphs clip the edges of obstacles);
- the width and trafficability of corridors are typically ignored;
- distance is generally the only factor considered in choosing the optimal path.

In the **vertex graph approach**, only the endpoints (vertices) of possible path segments are represented [15].

Advantages:

– this approach is suitable for spaces that have sufficient obstacles to determine the endpoints.

Disadvantages:

– determining the vertices in "open" terrain is difficult;

– trafficability over the path segment is not represented;

– factors other than distance cannot be included in evaluating possible routes.

In the **potential field approach**, the goal (destination) is represented as an "attractor", obstacles are represented by "repellors", and the vehicles are pulled toward the goal while being repelled from the obstacles.
Disadvantages:

– the vehicles can be attracted into box canyons from which they cannot escape;

– some elements of the terrain may simultaneously attract and repel.

In the **regular grid approach**, a grid overlays the terrain, terrain features are abstracted into the grid, and the grid rather than the terrain is analyzed.
Advantages:

– simplification of the analysis.

Disadvantages:

– "jagged" paths are produced because movement out of a grid cell is restricted to four (or eight) directions corresponding to the four (or eight) neighboring cells;

– granularity (size of the grid cells) determines the accuracy of terrain representation.

A many of route planners in the literature are based on the Dijkstra's shortest path algorithm, A* algorithm [7], geometric path planning algorithms [15] or its variants [12, 13, 18, 26, 31, 32]. For example, A* has been used in a number of computer generated forces systems as the basis of their planning component, to plan road routes [3], avoid moving obstacles [10], avoid static obstacles [18] and to plan concealed routes [14]. Very extensive discussion related to geometric shortest path planning algorithms was presented by Mitchell in [15] (references consist of 393 papers and handbooks). Geometric shortest paths problem is defined as follows: given a collection of obstacles, find an Euclidean shortest obstacle-avoiding path between two given points. Mitchell considers following problems:

– geodesic paths in a simple polygon;

– paths in a polygonal domain (searching the visibility graph, continuous Dijkstra algorithm);

– shortest paths in other metrics (L_p metric, link distance, weighted region metric, minimum-time paths, curvature-constrained shortest paths, optimal motion of non-point robots, multiple criteria optimal paths, sailor's problem, maximum concealment path problem, minimum total turn problem, fuel-consuming problem, shortest paths problem in an arrangement);

– on-line algorithms and navigation without map;

– shortest paths in higher dimensions.

3. Effectiveness problems in route planning

We focus one's attention on path planning algorithms and its effectiveness. Path planning algorithms used in battlefield simulation can be off-line or on-line. Off-line path planning algorithms like A* or Dijkstra's algorithm (listed in the previous section) find the whole solution before starting execution (simulation). They plan paths in advance and usually find optimal solutions. Their efficiency is not considered to be crucial and the moved object just follows the generated path. Although this is a good solution for a static environment, it is rather infeasible for dynamic environments, because if the environment or the cost functions are changed, the remaining path may need to be replanned, which is not efficient for real-time applications (e.g. real-time simulation). Let's recall that standard Dijkstra's algorithm has time complexity $O(V^2)$, where V denotes number of nodes in the graph. This complexity may be improved (if the graph is thin) implementing priority queue as binary heap, obtaining $O(E \cdot \lg V)$, or implementing priority queue as Fibonacci heap, obtaining $O(E + V \cdot \lg V)$, where E describes number of graph's edges.
In Fig. 4 we have graph of calculations time for single shortest path problem using standard Dijkstra algorithm in regular grid network with V nodes[1] (each path was calculated for the left-lower and the right-upper pair of cells (nodes) in grid network (similar to one from Fig. 6).
In Fig. 5 we have graph of calculations time for the same problem but defined as linear programming problem and solved using GAMS solver. From Fig. 4 results that when we must compute shortest path in grid network with e.g. $V = 400$ nodes (grid with size 20×20) then computational time is about 100 ms (for average case) using 1 GFLOPS processor and Dijkstra's algorithm. Let's suppose that we simulate battlefield for two-sided company level on the terrain area with size 16 km^2 (terrain square with 4×4 km size, so 4 km/20 = 200 m is side length for each of 400 cells). If we assume, that each company has 3 platoons then in the same simulation time we must plan movement, in the worst case, for $2 \times 3 = 6$ platoons (as non-divided

[1]Using computer with 1 GFLOPS processor (like PENTIUM III 800 MHz).

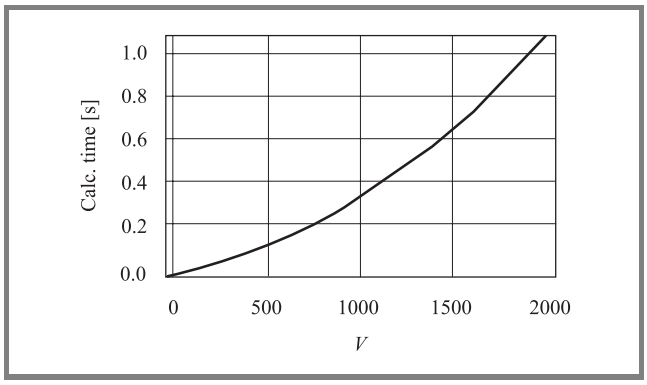

Fig. 4. Calculations time for single shortest path problem using Dijkstra algorithm in regular grid network with V nodes.

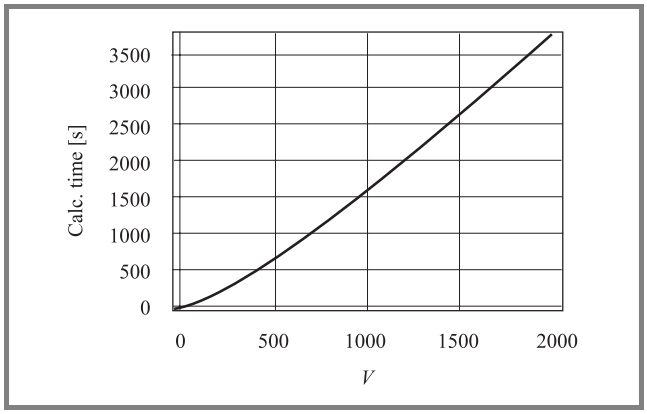

Fig. 5. Calculations time for single shortest path problem defined as linear programming problem and solved using GAMS solver in regular grid network with V nodes.

objects). Because these calculations must be done sequentially (having single processor), so estimation of computational time for all objects is about 6×100 ms = 600 ms. In this case we assumed that all processor power is used for path planning algorithm but it is some simplification, of course. Having, i.e. two-sided battalion fighting ($2 \times 3 \times 3 = 18$ platoons to plan movement in the same simulation time, in the worst case) we need 18×100 ms ≈ 2 s. This delay has significant effect on smoothness of simulation and its visualisation. And we should take into considerations that the network with 20×20 cells is small from among needed in battlefield simulation process.

There are three ways to increase effectiveness of considered problems:

- decreasing the size of terrain-based graph to decrease the computational time of paths planning algorithms [1];
- using specific on-line paths planning algorithms [10, 12, 13, 16, 32];
- using some partial path update approaches [23, 26].

Each of mentioned above ways has some advantages and disadvantages.

Advantage of the first way (**decreasing the size of graph**) is that the number of merged cells into regions will, in general, be far fewer than the number of grid cells. The disadvantages include difficulty in defining the center of the region and the computation difficulties in determining the optimum paths between two adjacent cells.

Some partial path planning algorithms (the second way) plan an off-line path, let the object follow the path, and if any new environment information is gathered, they partially re-plan the existing solution. Similar approach for multi-convoy redeployment in stochastic, dynamically changed environment, was presented in [26, 27]. Disadvantage of this approach is that some times, a small change in the environment may cause re-plan almost a complete path, which may take a long process time (when the network size is big).

The basic idea of **on-line path planning approach** (the third way), in generally, is that the object is moved step-by-step from cell to cell using some heuristic method. This approach is borrowed from movement robots path planning [13, 23, 32]. The decision about the next move (its direction, speed, etc.) depends on the current location of the object and environment status. For example, the idea of RTEF (real-time edge follow) algorithm [32] is to let the object eliminate closed directions (the directions that cannot reach the target point) in order to decide on which way to go (open directions). For instance, if the object has a chance to realize that moving to north and east will not let him reach the goal state, then it will prefer going to south or west. RTEF find out these open and closed directions, so decreasing the number of choices the object has. However, this approach has one basic disadvantage. Namely, in this approach using a few criterions simultaneously to find optimal (or acceptable) path is difficult and it is rather not possible to estimate, in advance, moment of achievement the destination. Moreover, it does not guarantee finding optimal solutions and even suboptimal ones may significanly differ from acceptable.

From this cause, we present in the next section hybrid, cells-merging-based and partial path planning approach for route planning in dynamically changed environment.

Considering route planning in the battlefield simulation we must mention multi-convoy (or multi-object) redeployment and, in consequence, multi-paths planning. Complexity of this process depends on the following conditions [29]:

- count of objects in each convoy (the convoy longer the scheduling of redeployment more complicated);
- have convoys be redeployed simultaneously?
- can convoys be destroyed during redeployment?
- can terrain-based network be destroyed during redeployment?
- have convoys be redeployed through disjoint routes?
- have convoys achieve selected places (nodes) at fixed time?
- do convoys have to start at the same time?

– have convoys determine any action strips for moving?
– can convoys be joined and separated during redeployment?
– have convoys cross through fixed nodes?, etc.

The most often problem related to multi-convoy redeployment is to move a few convoys through disjoint paths simultaneously [25, 30]. Disjoint paths condition results from safety ensuring for moved convoys. In the battlefield simulation finding disjoint paths for moved objects (e.g. tanks inside tank platoon) simplifies its movement because route for each tank do not cross route for each other and we avoid potential collisions. Disjoint paths optimization problem is NP-hard, so some heuristic or other suboptimal approaches are used [2, 21, 25, 31]. Description of some prototype module for maneuvre planning using disjoint paths approach was presented in [28].

4. A new multiresolution approach for increasing route planning effectiveness

Discussed, in the previous section, region approach for terrain included difficulty in defining the center of the region and the computational difficulties in determining the optimum paths between two adjacent cells. In this section we propose some multiresolution-based approach for finding shortest paths in the big grid networks. We assume that we have grid graph $G = \langle \boldsymbol{V}, \boldsymbol{A} \rangle$ (see Fig. 7) as representation of terrain squares (see Fig. 6), where $\boldsymbol{V}$ describes set of

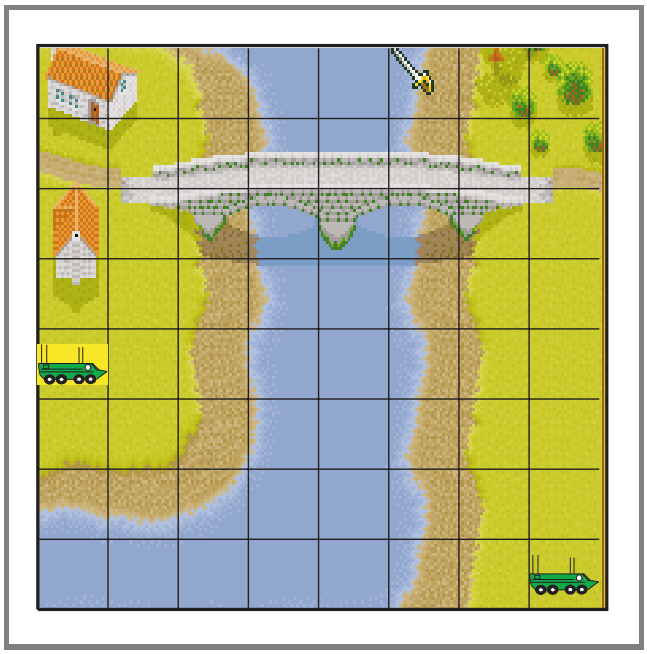

Fig. 6. Terrain space with division on regular grid squares. We want to move object from the right-lower corner to the left side.

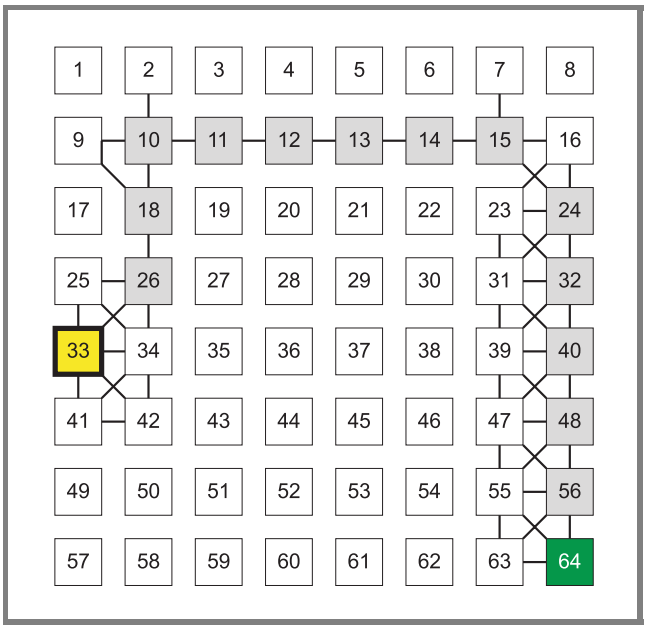

Fig. 7. Grid graph as representation of terrain squares from Fig. 6. There is marked shortest path from node 64 to node 33.

nodes (squares of terrain), $V = |\boldsymbol{V}|$, $\boldsymbol{A}$ describes set of arcs, $\boldsymbol{A} = \{\langle x,y \rangle \subset \boldsymbol{V} \times \boldsymbol{V} : \text{square } x \text{ is adjacent to square } y\}$.
In this graph we may describe some functions (as traversability, visibility, crossing time, crossing probability, detecting probability, etc.) obtaining network as model of movement environment. We assume that for each arc $\langle x,y \rangle \in \boldsymbol{A}$ we have cost $c(x,y)$. The idea of the approach is to merge geographically adjacent small squares (nodes belonging to $\boldsymbol{V}$) into bigger squares (called b-nodes, see Fig. 8) and build b-graph $\overline{G}$ (graph based on the b-nodes, see Fig. 9) using specific transformation. This transformation is based on the assumption that we set arc (b-arc) between two b-nodes $\overline{x} \subset \boldsymbol{V}$, $\overline{y} \subset \boldsymbol{V}$ when exist such two nodes $x \in \overline{x}$, $y \in \overline{y}$ that $\langle x, y \rangle \in \boldsymbol{A}$. In practice, as nodes of $\overline{G}$ graph we will consider strongly connected components of b-nodes. Cost $\overline{c}(\overline{x}, \overline{y})$ of the b-arc $\langle \overline{x}, \overline{y} \rangle \in \overline{\boldsymbol{A}}$ is set on the basis of the biggest cost of some shortest paths calculated inside the subgraph built on the nodes of $\overline{x}$. Next, in the b-graph we find shortest paths between such pairs $\overline{x_s}, \overline{y_t}$ of the b-nodes that source node s and target node t belong to sets $\overline{x_s}, \overline{y_t}$, respectively.
Formal definition of the graph $\overline{G}$ is as follows:

$$\overline{G} = \langle \overline{\boldsymbol{V}}, \overline{\boldsymbol{A}} \rangle, \tag{1}$$

where:

$$\overline{\boldsymbol{V}} = \{\overline{x}_1, \overline{x}_2, \ldots, \overline{x}_n\}\text{-set of b-nodes, } |\overline{\boldsymbol{V}}| = n,$$

$$\overline{x}_i = \{x_{i1}, x_{i2}, \ldots, x_{im}\} \subset \boldsymbol{V},\ i = \overline{1,n},$$

$$\underset{\substack{i,j \\ i \neq j}}{\forall} \overline{x}_i \cap \overline{x}_j = \emptyset,\ i = \overline{1,n},\ j = \overline{1,n},\ \bigcup_{i=1}^{n} \overline{x}_i = \boldsymbol{V},$$

$$\overline{\boldsymbol{A}} = \left\{ \langle \overline{x}, \overline{y} \rangle \subset \overline{\boldsymbol{V}} \times \overline{\boldsymbol{V}} : \underset{x \in \overline{x}, y \in \overline{y}}{\exists} \langle x, y \rangle \in \boldsymbol{A} \right\}.$$

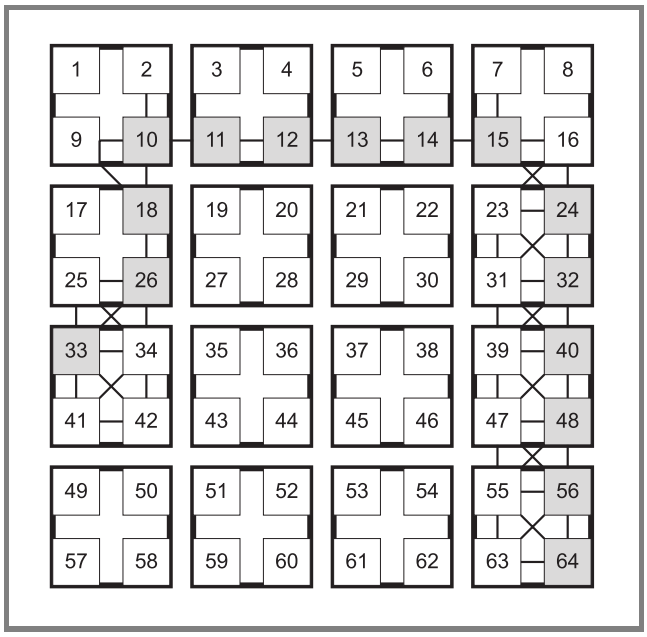

Fig. 8. Merging geographically adjacent small squares from Fig. 7 into $n = 16$ bigger squares (b-nodes).

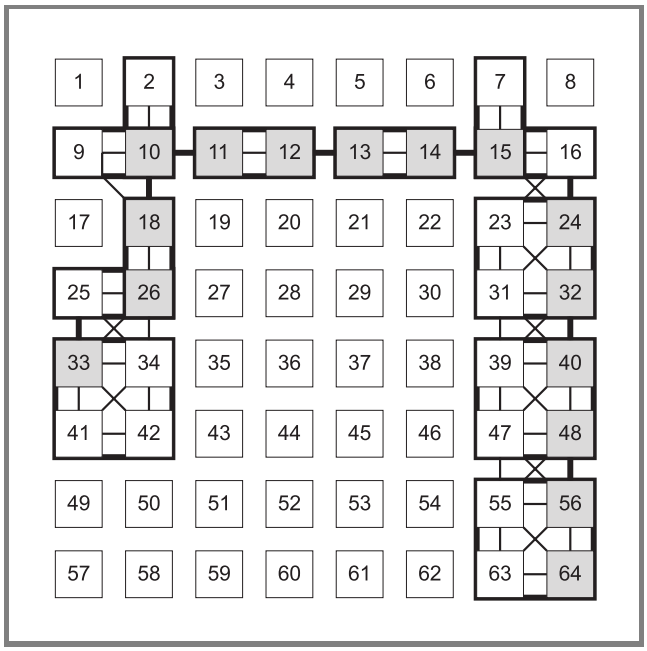

Fig. 9. b-graph for squares merging from Fig. 8. As b-nodes we use strongly connected components of b-nodes from Fig. 8.

Cost function $\overline{c}(\overline{x},\overline{y})$ for b-arc $\langle\overline{x},\overline{y}\rangle$ we determine as:

$$\overline{c}(\overline{x},\overline{y}) = \max_{\{x\in\overline{x}\}} F(x,\overline{y}), \tag{2}$$

where:

$$F(x,\overline{y}) = \min_{\left\{y\in\overline{y}:\ \exists\limits_{z\in\overline{x}}\langle z,y\rangle\in\boldsymbol{A}\right\}} L\big(P(x,y)\big),$$

$$L\big(P(x,y)\big) = \sum_{i=0}^{l(P(x,y))-1} c(x_i,x_{i+1}),$$

$$P(x,y) = (x_0 = x, x_1, x_2, \ldots, x_{l(P(x,y))} = y),$$

$$\forall_{i=\overline{0,l(P(x,y))-1}} \langle x_i, x_{i+1}\rangle \in \boldsymbol{A}.$$

The merging algorithm for b-graph-based shortest paths planning (MSP-algorithm) is following:

1. merge small squares from graph G (Fig. 7) into n bigger squares (Fig. 8) (n is parameter of the algorithm; we show in further discussion how we can set the optimal value of the n);
2. inside each of the n big squares (b-nodes) determine strongly connected components obtaining at least n subgraphs;
3. set each of subgraphs obtained from the Step 2 as b-nodes and arcs as described by (1) obtaining graph $\overline{G}$ (Fig. 9);
4. find shortest paths between each pair of nodes inside each b-node (subgraph) of $\overline{G}$ to calculate cost $\overline{c}(\overline{x},\overline{y})$ for each arc of $\overline{G}$ using Eq. (2);
5. find shortest path in $\overline{G}$ with cost function $\overline{c}(\cdot,\cdot)$ between such pairs $\overline{x_s},\overline{y_t}$ of b-nodes that source node s and target node t belong to sets $\overline{x_s},\overline{y_t}$, respectively.

It's important to explain that setting in the Step 3 strongly connected components as b-nodes assure that each node inside such component is attainable from each other, so if b-node $\overline{x}$ is connected (through b-arc) with b-node $\overline{y}$ then exist path from each node of $\overline{x}$ to each node of $\overline{y}$.
Let's estimate time complexity of MSP algorithm. We will estimate complexity of each step of the algorithm as follows (we assume that each b-node is strongly connected):

2. determination of strongly connected components in graph G: we have n b-nodes creating n merged subgraphs of G; each subgraph of G has no more than $\lceil\frac{V}{n}\rceil$ nodes, so we have complexity $O\left(n\cdot\lceil\frac{V}{n}\rceil\right) = = O(V)$;
3. we have n b-nodes so we obtain $O(n)$;
4. shortest path problem between each pair nodes in N-nodes graph has complexity $O(N^3)$; if each subgraph of G is strongly connected component of G, then has n b-nodes (creating subgraphs), so each subgraph has $\lceil\frac{V}{n}\rceil$ nodes, hence finding all-pairs shortest paths in single subgraph has complexity $O\left(\left(\frac{V}{n}\right)^3\right)$; because we must calculate it n times, so we have $O\left(n\cdot\left(\frac{V}{n}\right)^3\right)$;
5. finding shortest paths in graph $\overline{G}$: because $\overline{G}$ has n b-nodes, so using standard Dijkstra's shortest path algorithm we have $O(n^2)$.

We omit the merging Step 1 because, having n, we can prepare this step before simulation. Taking into considerations above estimations we obtain total complexity of the

algorithm as $O\left(\frac{V^3}{n^2}+n^2+V\right)$ (we have also omitted $O(n)$ because $n \ll V$).

There is very interesting and important question from the point of view of proposed approach effectiveness: how should we set n to obtain the better effectiveness than for V?

Let's notice that computational complexity of the algorithm based on the network with small squares[2] is $O(V^2)$ and for the algorithm based on the bigger squares is $O\left(\frac{V^3}{n^2}+n^2+V\right)$. It means that, in sense of complexity symbol $O(\cdot)$, the bigger squares approach is better if the following formula is satisfied:

$$\frac{V^3}{n^2}+n^2+V < V^2 \tag{3}$$

or equivalently, when

$$n^4-(V^2+V)n^2+V^3<0\,. \tag{4}$$

Solving this inequality we obtain, that $n \in [n_1, n_2]$, where

$$n_1=\sqrt{\frac{V^2+V-\sqrt{(V^2+V)^2-4V^3}}{2}}\,, \tag{5}$$

$$n_2=\sqrt{\frac{V^2+V+\sqrt{(V^2+V)^2-4V^3}}{2}}\,. \tag{6}$$

For example, for the graph from Fig. 7 ($V=64$) we obtain $n_1 \approx 8$, $n_2 \approx 64$.

In order to estimate how many times faster we compute problem for finding shortest path in the network with n big squares ($\left(\frac{V^3}{n^2}+n^2+V\right)$) with relation to problem for finding shortest path in the network with V small squares ($O(V^2)$) we may formulate acceleration function as follows[3]:

$$A(V,n)=\frac{V^2}{\frac{V^3}{n^2}+n^2+V}\,. \tag{7}$$

Exemplified graphs of $A(V,n)$ are shown in Figs. 10 and 11.

Having grid network with V squares (nodes) we can formulate following optimization problem: to find such cardinal n^*, for which

$$A(V,n^*)=\max_{n\in[n_1,n_2]} A(V,n)\,, \tag{8}$$

where n_1, n_2 are described by formulas (5) and (6).

[2]Using standard Dijkstra's shortest paths algorithm (without modifications increasing its effectiveness).

[3]Exact to complexity estimation symbol $O(\cdot)$.

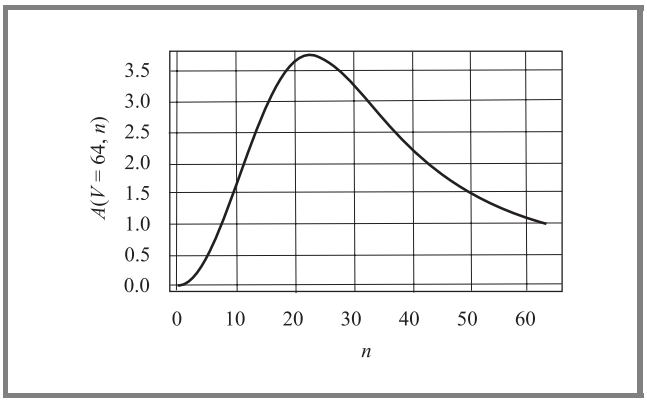

Fig. 10. Graph of $A(V, n)$ function for the network with $V = 64$ nodes.

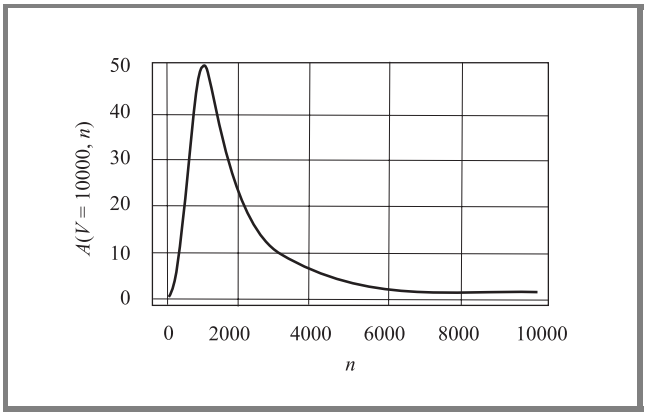

Fig. 11. Graph of $A(V, n)$ function for the network with $V = 10000$ nodes.

Let's notice, that function (7), omitting constraint for n integer, has real nonnegative maximum for the value $n^*=\sqrt[4]{V^3}$. It may be easily shown that for each $V>0$, $n^* \in [n_1, n_2]$. In practice we are interested in such value $n^{**} \approx n^*$, that square root of $\frac{V}{n^{**}}$ is cardinal number (it results from the fact that each of n^{**} big squares consist of $\frac{V}{n^{**}}$ small squares and in grid structure of the network a big square has $\sqrt{\frac{V}{n^{**}}} \times \sqrt{\frac{V}{n^{**}}}$ small squares).

In Table 1 the influence of V on n^*, n^{**} and $A(V,n^{**})$ is shown. It is easy to observe the best acceleration of shortest path algorithm using presented approach in regular grid network with V nodes may be approximated by value $A(V,n^{**}) \approx \frac{1}{2}\sqrt{V}$.

Let's notice that from presented estimations and Table 1 result that for 400-nodes grid graph considered at the beginning of the previous section movement planning for two-sided battalion fighting (for 18 platoons) will be done in time $18 \times 100/9.5$ ms ≈ 200 ms.

Table 1
Influence of V on n^*, n^{**} and $A(V,n^{**})$

V	n^*	n^{**}	$\lceil \frac{V}{n^{**}} \rceil$	$A(V,n^{**})$
100	32	25	4	4.3
400	90	100	4	9.5
900	164	225	4	12.3
1600	253	169	9	14.7
2500	354	256	9	20.4
10000	1000	1089	9	49.0
40000	2828	2500	16	96.8
90000	5196	5625	16	147.9
160000	8000	6400	25	181.4
250000	11180	10000	25	243.7

5. Conclusions

The approach presented in the paper gives possibilities to significantly decrease computational time in terrain-based route planning when the terrain environment is represented by regular grid of squares. This approach may be applied, i.e. for route planning in the simulated battlefield.

The estimations of presented algorithm effectiveness may be improved through a few ways. The first way is to use in time complexity estimations the best known shortest-path algorithm estimation $(O(E + V \cdot \lg V))$ instead complexity of standard Dijkstra's algorithm $(O(V^2))$ because the regular grid graph is thin (maximal number of direct successors for any node is 8), so $O(8V + V \lg V) < O(V^2)$ nearly for all V (exactly for $V > 11$). The second way is to improve Step 4 of the algorithm because it seems to be unnecessary determinations all-pairs shortest paths in each b-nodes (subgraphs). It's seems that is enough to determine shortest paths between "outside" nodes of b-nodes because only these nodes are used to link b-node with another. Moreover, to confirm presented estimations it is essential to conduct calculations in real grid graphs.

Presented suggestions may be contribution for further works.

References

[1] J. R. Benton, S. S Iyengar, W. Deng, N. Brener, and V. S. Subrahmanian, "Tactical route planning: new algorithms for decomposing the map", in *Proc. IEEE Int. Conf. Tools for AI*, Herndon, 1995, pp. 268–277.

[2] A. Bley, "On the complexity of vertex-disjoint length-restricted path problems", Konrad-Zuse-Zentrum fur Informationstechnik, Berlin, 1998 (see also: http://www.zib.de/PaperWeb/abstracts/SC-98-20/).

[3] C. Campbell, R. Hull, E. Root, and L. Jackson, "Route planning in CCTT", in *Proc. 5th Conf. Comput. Gener. Forc. Behav. Repres.*, Tech. Rep., Institute for Simulation and Training, 1995, pp. 233–244.

[4] C. G. Cassandras, C. G. Panayiotou, G. Diehl, W.-B. Gong, Z. Liu, and C. Zou, "Clustering methods for multi-resolution simulation modeling", in *Proc. Conf. Enabl. Technol. Simul. Sci., Int. Soc. Opt. Eng.*, Orlando, USA, 2000, pp. 37–48.

[5] C. Cooper, A. Frieze, K. Melhorn, and V. Priebe, "Average-case complexity of shortest-paths problems in the vertex-potential model", *Rand. Struct. Algor.*, vol. 16, pp. 33–46, 2000.

[6] P. K. Davis, J. H. Bigelow, and J. McEver, "Informing and calibrating a multiresolution exploratory analysis model with high resolution simulation: the interdiction problem as a case history", in *Proc. 2000 Winter Simul. Conf.*, 2000, pp. 316–325.

[7] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths", *IEEE Trans. Syst. Sci. Cybern.*, vol. SSC-4, no. 2, pp. 100–107, 1968.

[8] J. James, B. Sayrs, J. Benton, and V. S. Subrahmanian, "Uncertainty management: keeping battlespace visualization honest", http://citeseer.nj.nec.com/ 386770.html

[9] L. Joe and P. M. Feldman, "Fundamental research policy for the digital battlefield", Res. Rep. DB-245-A, RAND Co., Santa Monica, USA, 1998.

[10] C. R. Karr, M. A. Craft, and J. E. Cisneros, "Dynamic obstacle avoidance", in *Proc. Conf. Distrib. Interact. Simul. Syst. Simul. Train. Aerosp. Envir., Int. Soc. Opt. Eng.*, Orlando, USA, 1995, pp. 195–219.

[11] T. Kreitzberg, T. Barragy, and B. Nevin, "Tactical movement analyzer: a battlefield mobility tool", in *Proc. 4th Join Tactic. Fus. Symp.*, Laurel, 1990.

[12] B. Logan, "Route planning with ordered constraints", in *Proc. 16th Works. UK Plann. Schedul. Spec. Int. Group*, Durham, UK, 1997.

[13] B. Logan and A. Sloman, "Agent route planning in complex terrains", Tech. Rep. CSRP-97-30, University of Birmingham, School of Computer Science, Birmingham, 1997.

[14] M. Longtin and D. Megherbi, "Concealed routes in ModSAF", in *Proc. 5th Conf. Comput. Gener. Forc. Behav. Repres.*, Tech. Rep., Institute for Simulation and Training, 1995, pp. 305–314.

[15] J. S. B. Mitchell, "Geometric shortest paths and network optimization", in *Handbook of Computational Geometry*, J. R. Sack and J. Urrutia. Elsevier Science Publ., B.V. North-Holland, Amsterdam, 1999.

[16] D. K. Pai and L. M. Reissell, "Multiresolution rough terrain motion planning", Department of Computer Sciences, University of British Columbia, Tech. Rep. TR 94-33, Vancouver, 1994.

[17] M. D. Petty, "Computer generated forces in distributed interactive simulation", in *Proc. Conf. Distrib. Interact. Simul. Syst. Simul. Train. Aerosp. Envir., Int. Soc. Opt. Eng.*, Orlando, USA, 1995, pp. 251–280.

[18] S. Rajput and C. Karr, "Unit route planning", Tech. Rep. IST-TR-94-42, Institute for Simulation and Training, Orlando, USA, 1994.

[19] G. A. Schiavone, R. S. Nelson, and K. C. Hardis, "Interoperability issues for terrain databases in distributed interactive simulation", in *Proc. Conf. Distrib. Interact. Simul. Syst. Simul. Train. Aerosp. Envir., Int. Soc. Opt. Eng.*, Orlando, USA, 1995, pp. 89–120.

[20] G. A. Schiavone, R. S. Nelson, and K. C. Hardis, "Two surface simplification algorithms for polygonal terrain with integrated road features", in *Proc. Conf. Enabl. Technol. Simul. Sci., Int. Soc. Opt. Eng.*, Orlando, USA, 2000, pp. 221–229.

[21] A. Schrijver and P. Seymour, "Disjoint paths in a planar graph – a general theorem", *SIAM J. Discr. Math.*, no. 5, pp. 112–116, 1992.

[22] H. Sherali, K. Ozbay, and S. Subrahmanian, "The time-dependent shortest pair of disjoint paths problem: complexity, models and algorithms", *Networks*, no. 31, pp. 259–272, 1998.

[23] A. Stentz, "Optimal and efficient path planning for partially-known environments", in *Proc. IEEE Int. Conf. Robot. Automat., ICRA'94*, vol. 4, pp. 3310–3317.

[24] P. D. Stroud and R. C. Gordon, "Automated military unit identification in battlefield simulation", LAUR-97-849, *SPIE Proc.*, vol. 3069, Los Alamos National Laboratory, Los Alamos, 1997.

[25] Z. Tarapata, "Algorithm for simultaneous finding a few independent shortest paths", in *Proc. 9th Eur. Simul. Symp., ESS'97, Soc. Comput. Simul.*, Passau, Germany, 1997, pp. 89–93.

[26] Z. Tarapata, "Simulation method of aiding and estimation of transportation columns movement planning in stochastic environment", in *Proc. 13th Eur. Simul. Multiconf., Soc. Comput. Simul. Int.*, Warsaw, Poland, 1999, pp. 613–619.

[27] Z. Tarapata, "Computer simulation of individual and grouped military objects redeployment", *Bull. Milit. Univ. Technol.*, no. 1, pp. 147–162, 2000.

[28] Z. Tarapata, "Computer tool for supporting and evaluating convoys redeployment planning", *Oper. Res. Decis.*, no. 1, pp. 91–107, 2000.

[29] Z. Tarapata, "Some aspects of multi-convoy redeployment modelling and simulation", in *Proc. 21st AFCEA Eur. Symp. & Exposit.*, Prague, 2000 (compact disk publication).

[30] Z. Tarapata, "Modelling, optimisation and simulation of groups movement according to group pattern in multiresolution terrain-based grid network", in *Proc. Reg. Conf. Milit. Commun. Inform. Syst.*, Zegrze, Poland, 2001, vol. I, pp. 241–251.

[31] Z. Tarapata, "Fast method for redeploying multi-convoy in multiresolution grid network", *Bull. Milit. Univ. Technol.*, 2003 (in press).

[32] C. Undeger, F. Polat, and Z. Ipekkan, "Real-time edge follow: a new paradigm to real-time path search", SCS Publications, 2001 (see also: http://citeseer.nj.nec.com/489498.html).

Zbigniew Tarapata has graduated from Cybernetics Faculty at Military University of Technology (MUT) in Warsaw. He received his Master's degree in computer science in 1995 and Doctor's degree in the same field, in 1998. From 1995 to July 1999 he worked as an assistant and since July 1999 – as a senior lecturer at Operations Research Division of Institute of Mathematics and Operations Research of Cybernetics Faculty of MUT. His scientific interests and work are related to the following subjects: mathematical and simulation modelling of systems; transport optimization; combat modelling, simulation, optimization and prediction; graph and network optimization; algorithms effectiveness; methods and tools of multicriteria decision making and supporting.
e-mail: ztarap@isi.wat.waw.pl
Institute of Mathematics and Operations Research
Faculty of Cybernetics
Military University of Technology
Kaliskiego st 2
00-908 Warsaw, Poland