O^3: An Optimal and Opportunistic Path Planner (with Obstacle Avoidance) using Voronoi Polygons

David M. Coleman and Joseph T. Wunderlich, PhD.
Elizabethtown College, Elizabethtown, Pennsylvania, USA
{colemand, wunderjt}@etown.edu

Abstract- **Traditional mobile robot research focuses on a robot navigating its environment to reach a single goal while avoiding obstacles. This paper proposes a new method called O^3 to solve the challenges presented at the Intelligent Ground Vehicle Competition (IGVC) where a navigation course includes multiple goals to be found in an optimal order. The O^3 technique includes improvements on traditional path planning and obstacle avoidance techniques while providing an explicit ability to change course as obstacles are discovered. This method uses modern trajectories such as minimum-weighted Hamiltonian circuits, A* algorithm for obstacle avoidance, and local points of opportunity to update the globally optimal path using Voronoi polygons. Environmental mapping is also used to speed up the search algorithms in static environments. Overall, the O^3 technique exploits local points of opportunity while avoiding obstacles and ultimately finding a globally optimal path through an unknown environment.**

This methodology will be implemented on an autonomous web-based tour guide robot to serve the Internet community reviewing Elizabethtown College. This methodology can be extended to other research areas where multiple locations need to be traversed independent of their order such as city map, trip planners, and distribution networks (power, internet, etc) due to its balance between weighted graphs and obstacle avoidance (objects, traffic, construction, etc).

I. Introduction

Mobile robotic motion control can be separated into two research areas: (1) simple obstacle-free path planning and (2) path planning which includes various obstacle avoidance strategies. An example of a simple obstacle-free path planning strategy is creating a Hamiltonian circuit through a set of given nodes [10]. *A priori* information may be added to plan a specific path around an obstacle [4]. However, obstacle locations are often not known ahead of exploration by the robot; this warrants developing more complex obstacle avoidance strategies.

In [1], a dynamic window approach provides a "local vs. global" relationship and is used to store obstacles in memory for later analysis. Heuristics along with additional feedback from sensors are used to provide motion and obstacle locations as seen in [9]. To improve the ease of mapping it is suggested in [3] that obstacles be scaled to match the dimensions of the robot. Other obstacle avoidance decisions are done using the cell-decomposition methods of VFH* [4] and Sentz's A* algorithm [7].

Most importantly the O^3 technique heavily relies on Voronoi diagrams. The Voronoi diagram is used in other path planners including the roadmap [14] and HGVG algorithms [5]. A formal definition of the Voronoi diagram can be found in [5], [14], and [16]. Related to the Voronoi diagram is the Delaunay triangulation. As defined in [16], the Delaunay triangulation T is the maximum planar subdivision of n points $P = \{P_1 \ldots P_n\}$ such that no points of P are bounded by the circumcircle of any triangle in T. The Delaunay triangulation will be used to restrict the domain of our algorithms and will be expanded upon throughout this paper. Examples of each of these are shown in Fig. 1(b), (c), respectively.

II. Path Planning

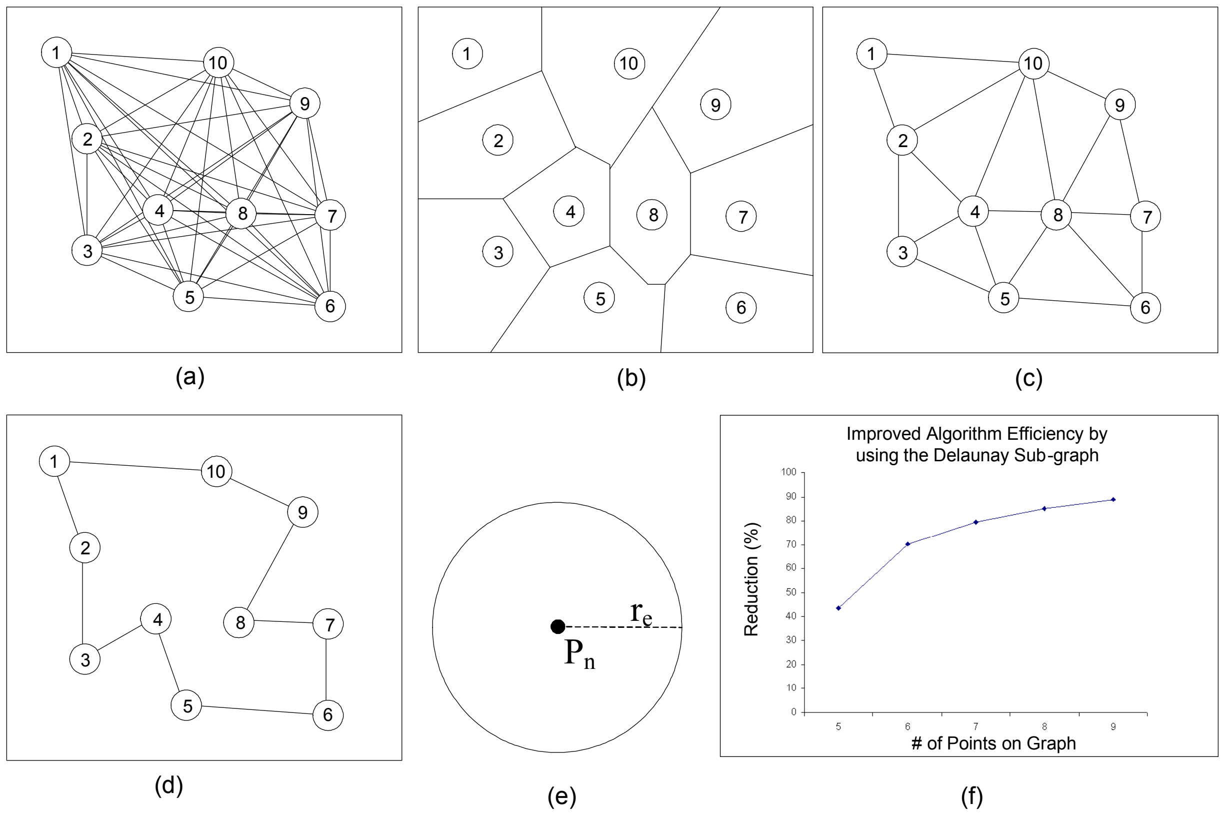
As stated in [10] and [14] a path planner must be correct and complete. Correct meaning the algorithm must be accurate and complete meaning an algorithm must return a failed value when a solution is not available in a reasonable amount of time. This is a requirement for both explicit (before motion) and implicit (during motion) methods.

A. Classical Approach – Explicit Methods

Example: Imagine walking into a room for the first time with the lights off and being asked to find the door on the opposite side of the room. You do not process any knowledge of the room's exact dimensions, obstacles, or the quickest path to the door. However, having the knowledge that a door exists (or assuming it exists), and knowledge that its location is approximately "across" the room, is enough information to plan a path (including an obstacle-avoidance strategy) – even though it may not be optimal.

Similarly, a robot enters an unknown environment and the complexity of tasks is increased with multiple nodes to be visited and exact distances and velocities to be rendered. The information given for target nodes may include GPS position. Thus specific distances can be calculated based on the entrance point of the robot; and since the end of the overall path plan often includes returning to the entrance, this often becomes a typical Traveling Salesman Problem (TSP) [11]. By solving the TSP for the given set, a solution will exist that is minimal in distance (and likely to be traversable by the robot).

A breadth-first search through all possible Hamiltonian circuits is logically the easiest, but most difficult computationally. Given a set of n points, as shown in Fig. 1(a),



(a)
(b)
(c)
(d)
(e)
(f)
r_e
P_n
Improved Algorithm Efficiency by using the Delaunay Sub-graph
Reduction (%)
# of Points on Graph

Fig. 1. (a) Environment of 10 goal nodes; (b) Voronoi Diagram of environment; (c) Delaunay Triangulation of environment; (d) Hamiltonian circuit derived by (c); (e) Expanded node showing r_e (f) Reduction in processing time using (c) instead of (a).

finding the TSP solution through a breadth-first search is on the order

$$O((n-1)!) \tag{1}$$

since every point has a degree of (n-1) and every point can be reached from any starting point. A general source code structure would look like the following:

```
% Input node locations
% Define adjacency matrix
Set (control_flags)
for i = 1…(n-1)!
     path = get(g, Hamiltonian);
     if ( is_unique(path) )
          P = path(information);
     end
     Reset control flags
end
return shortest(p)
```

Given the Voronoi diagram in Fig. 1(b) the Delaunay triangulation graph can be constructed by connecting points within touching Voronoi polygons and is shown in Fig. 1(c). By Euler's formula (V-E+F=2) [15] the sub-graph is said to have at most 3n-6 edges and an extreme less number of Hamiltonian paths. Testing has shown a breadth-first search of the Delaunay graph is on the order

$$O(n) \tag{2}$$

The actual number of Hamiltonian circuits that exist in the Delaunay sub-graph of any graph is dependant on the topology of the graph and therefore is not generally quantifiable. Introducing the Delaunay sub-graph into the previous code segment results has shown to improve the search for a TSP solution between 40% and 90%. The specific results and testing software will be explained later in this paper. Fig. 1(f) shows the improved algorithm efficiency and corresponds to Table 1.

Throughout this paper the term "TSP solution" will be used to identify the Hamiltonian circuit in the Delaunay domain. By previous arguments TSP solution can be said to be correct since the globally optimal solution was not lost in the Delaunay reduction. With the absence of obstacles and unreachable points within the graph the TSP solution and breadth-first algorithm are complete.

It can also be said that by using the TSP solution the robot will intersect the acceptable radius error (r_e) on each target

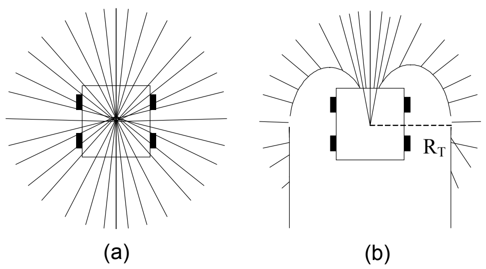


Fig. 2. Robot with (a) zero turning radii, (b) actual turning radii, RT

node. Once obstacles are introduced implicit methods are employed and the TSP solution is used as a governing overview path.

B. Implicit Methods

Recalling the thought experiment from the previous section, imagine starting to move in the direction you think is correct to reach the door. How do you go about detecting, avoiding, and overall re-evaluating the best path to follow to reach your goal? Do you just run into objects that may be in the room and bounce off, or do you feel with your arms and attempt to adjust your senses to the dark room? Let us also focus on achieving one goal and avoiding obstacles along the way.

A robot has many sensors that contribute to the "overview" of an environment. Feedback needs to be assessed in real-time to adjust the course of the robot. Two things need to be considered for path alteration: turning radius and size of the dynamic window as described in [13]. The specific algorithms for real-time obstacle avoidance are beyond the scope of this paper. Therefore, only key points will be discussed for completeness of the O^3 techniques.

As shown in Fig. 2(a), the easiest approach is to assume the robot is a point position and can change its course at any time. However, when implemented, this is not often the case as seen in [3].

As outlined in [1], a dynamic window is common among implicit algorithms for obstacle avoidance. The sensors available on the robot govern the exact dimension of this window. These can include laser range finders, sonar, machine vision, and magnetic/GPS positioning. An example of the window approach is seen in Fig. 3. Depending on the sensors used, only certain directions may need monitored (i.e. forward) and the previously explored areas can be stored in an environmental map. In static environments, a map in memory can be an effective tool to speed up processing time for revisited areas.

By using the dynamic window approach, heuristics can be

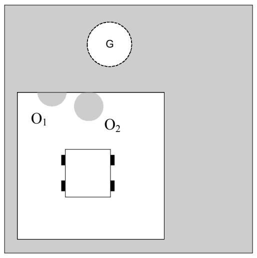


Fig. 3. Dynamic window approach. Obstacles are seen at O_1 and O_2 and the goal node is labeled G.

employed for implicit path planning and obstacle avoidance. As seen in Fig. 4, there are situations where a decision is not obvious with the limited view of the sensors. As stated in [4] "a larger trigger distance would not eliminate the problem." In fact it could increase the run time for the algorithm computing all possible avenues, and thus be incomplete. By using a heuristic approach such as A*, a decision based on a cost function will be made and followed as shown in path C of Fig. 4. This cost function is explained in [7].

C. Local Opportunistic / Globally Optimal Points

To conclude our previous thought experiment, now imagine moving towards one goal and, through sensory information, an obstacle blocks the intended explicitly defined path. While avoiding the obstacle, another point becomes more desirable for traversal in distance and availability. Therefore a change in course should be analyzed to see if it is in fact globally optimal and not just locally opportunistic.

This locally opportunistic globally optimal visit of an out-of-order node is a combination of the TSP solution developed in the first section and the obstacle/motion techniques developed in the second section of this paper. An obstacle is shown in Fig. 5(a), which interferes with the TSP solution explicitly planned. By avoiding the obstacle using techniques previously discussed, a new point becomes locally opportunistic.

Consider the simplified graph shown in Fig. 5(c). This graph is connected with edges {D1, D2, D3, T1, T2, T3} as paths. Assume the ideal case where the turning radii (r_r and r_l) are neglected and straight paths are possible. Also assume there are no obstacles in the paths between {C, 8},{7,8},{7,*n*},{8,*n*}. Let C be the point where the path crosses into the unexpected Voronoi polygon as shown to the left of the obstacle in Fig. 5(a) surrounding point 8. Allow point *n* to be the next point to be traversed after the set of three points shown.

Two equations can be shown: the current path as specified

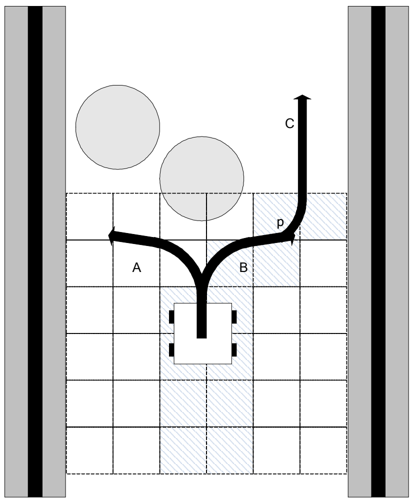


Fig. 4. Cell decomposition of region where one obstacle is in the middle of the robot's vision. A decision needs to be made to take path A (left curve) or path B (right curve). Using the heuristics invoked in A* path B is chosen and the best-fit curve (path C) is implemented at point P. The curvature at point P is equal to r_l. The grid surrounding the robot represents the environmental map being developed with A*. Its size is equal to the dynamic window shown in Fig. 3 and is govern by available sensors.

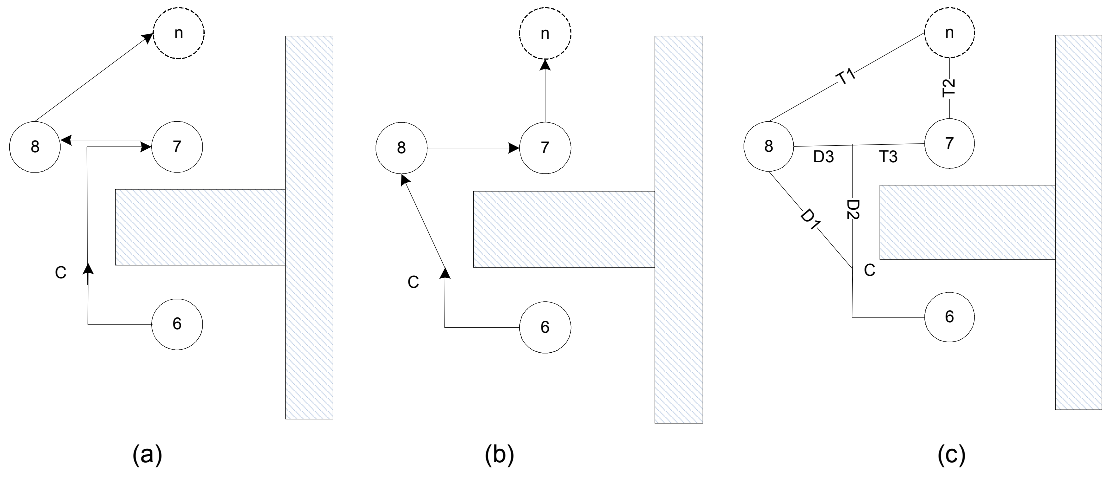


Fig. 5. Following the path prescribed in Figure 1(d) the robot should traverse the points as shown in (a). With the discovery of the obstacle it changes points to traverse the path shown in (b). (c) The combination of (a) and (b) for γ analysis.

by the TSP solution in Fig. 5(a):

$$\{6, 7, 8, n\} = (d_{6C}) + (D2+T3) + (T3+D3) + (T1) \quad (3)$$

and the opportunistic path shown in Fig. 5(b):

$$\{6, 8, 7, n\} = (d_{6C}) + (D1) + (D3+zT3) + (T2). \quad (4)$$

It must be shown that in order for the path to be optimal (not just ideally opportunistic) the new path eq. (4) is less weighted than the current TSP solution in (3).

$$(d_{6C}) + (D1) + (D3+T3) + (T2) < (d_{6C}) + (D2+T3) + (T3+D3) + (T1) \quad (5)$$

Subtracting (d_{6C}) from both sides:

$$(D1) + (D3+T3) + (T2) < (D2+T3) + (T3+D3) + (T1) \quad (6)$$

By trigonometry rules:

$$D1 = (D2^2 + D3^2)^{1/2} \text{ and } T1 = ((T3+D3)^2 + T2^2)^{1/2} \quad (7)$$

Combining (6) and (7):

$$((D2^2 + D3^2)^{1/2}) + (D3+T3) + (T2) < (D2+T3) + (T3+D3) + (((T3+D3)^2 + T2^2)^{1/2}) \quad (8)$$

Expanding and eliminating like terms:

$$\gamma = (D3^2+2*D3*T3+T2^2+T3^2)^{1/2} + T3 - T2 - (D2^2 + D3^2)^{1/2} + D2 \quad (9)$$

Therefore, when $\gamma > 0$ the alternative path is optimal and the course should be altered.

Once again it is essential that the local point-of-opportunity does not compromise the correctness and completeness of the global solution. Therefore it is necessary that the point of non-opportunity be the target point after the opportunity point. It is also sufficient in the case where multiple points-of-opportunity exists on the way to the original non-opportunistic point. This would solve the problem where a point of interests resides in an area of limited opening such as a box shown in Fig. 6.

In Fig. 6 the explicit TSP solution dictates an A-B-C-D path. However, after γ analysis, point C is determined to be locally opportunistic and globally optimal. Now an A-C-B-D solution exists. While on route to point B, point D is determined to be locally opportunistic and globally optimal. Once again the path is now changed to A-C-D-B. By this repetitive process the γ analysis will prove to be globally correct and complete. There is an assumption made that all points are reachable by the robot.

The γ analysis is a technique that is only necessary to perform when crossing over an unexpected Voronoi polygon. Similar to OPEN and CLOSED (and RAISED and LOWERED) sets in Sentz's A* algorithm a set needs to be created for Voronoi polygon crossings along the path between two points. By setting a flag for an unexpected Voronoi polygon crossing the γ analysis will be limited to only those of

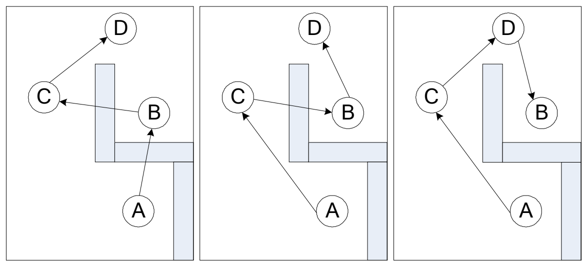


Fig. 6. Map with more than one point of opportunity. The original TSP dictates an A-B-C-D path. After the γ analysis, A-C-D-B is locally opportunistic and globally optimal.

Processing Time Comparison (in seconds)					
	Number of input points				
	Five	Six	Seven	Eight	Nine
Full graph	0.0330	0.3656	10.12	575.0	41570
Delaunay sub-graph	0.0183	0.1059	2.036	84.97	4547
Improvement (%)	43.68 ±20.42	70.31 ±7.422	79.29 ±3.915	85.01 ±3.016	89.06†

Table 1. Reduction of processing time shown by using the Delaunay sub-graph in Fig 1(c) over the complete connected graph of Fig. 1(a). †Standard deviation was not calculated, as only one test run was available due to the time interval required to complete one pass.

true opportunity points. Also, the polygonal crossings are used to speed up the "nearest" point question. By knowing specifically which Voronoi polygon the robot resides in, the nearest point is apparent.

By both these techniques (γ analysis and polygon sets) O^3 expands traditional robot navigation for a multiple-target environment while still employing traditional obstacle avoidance strategies.

III. Implementation

Our methods for path planning and obstacle avoidance are being implemented to solve the challenges at the IGVC competition in May 2008. One challenge is navigating a long, complex maze defined by white lines painted on approximately 3-inch tall grass, and riddled with various complex obstacles (e.g. ramps, pits, trees, fencing, and various cones). Using A* heuristics with dimensional constraints in the cost functions as outlined in [3] [4], both correct and complete decisions can be made. With the addition of environmental mapping in a static environment, obstacle-processing time can be minimal. In the second IGVC challenge of GPS navigation with minimal obstacle avoidance, our O^3 method is ideal. Our *Wunderbot 4* has laser ranges finders, vision system with color recognition, digital compass, GPS receiving, and optical encoders for sensor feedback.

IV. Preliminary Results

Table 1 shows the improved processing time seen in our code during multiple trials. Matlab R2007 was chosen to test our method because it supports graphing, provides functional toolboxes for efficient environmental geometry analyzing and straightforward integration into LabVIEW. LabVIEW provides the real-time operating controls necessary for the Wunderbot 4. The testing was done on an Intel Core Duo CPU @ 2.00GHz. Overall, the Delaunay sub-graph provided an average improvement over 73%.

V. Future Research

After competition, the *Wunderbot 4* will serve as a platform for autonomous web-driven tours on the Elizabethtown College campus. With our O^3 implemented, a pre-determined or "way-point" driven tour is not necessary. In fact, any path can be altered (or tailored) based on student traffic, construction paths, and/or availability of certain building/fields on campus. By changing the weights of certain paths in the TSP solution and cost function (with obstacles), a fully autonomous robot is possible with O^3.

VI. Conclusion

O^3 is a unique method that combines traditional implicit and explicit methods to offer an optimal and opportunistic (and obstacle avoidant) solution to the areas of path planning for autonomous robots. By continually implying graphical techniques such as Voronoi polygons short cuts in calculations can be achieved as well as opportunistic paths through an unknown environment. This methodology should also yield successful results for our Wunderbot 4 robot at international competition and as a robot tour-guide.

VII. ACKNOWLEDGMENTS

We would like to thank the math department at Elizabethtown College for their continuing support and collaboration of theories within this proceeding, especially Dr. Bobette Thorsen.

References

[1] O. Brock, O. Khatib, "High speed navigation using the global dynamic window approach", IEEE International Conference on Robotics and Automation, May 10-15, pp. 341-346 vol.1

[2] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots in cluttered environments", IEEE International Conference on Robotics and Automation, Cincinnati, Ohio, May 13-18, 1990, pp.572-577

[3] I. Ulrich and J. Borenstein, "VFH+: reliable obstacle avoidance for fast mobile robots", IEEE International Conference on Robotics and Automation, Leuven, Belgium, May 16-21, 1998, pp. 1572-1577

[4] I. Ulrich and J. Borenstein, "VFH*: local obstacle avoidance with look-ahead verification", IEEE International Conference on Robotics and Automation, San Francisco, CA, April 24-28, 2000, pp. 2505-2511

[5] H. Choset and J. Burdick, "Sensor-based exploration: the hierarchical generalized Voronoi graph", The International Journal of Robotics Research, Vol. 19, No. 2, February 2000, pp. 96-125

[6] A. Stentz, "Optimal and efficient path planning for partially-known environments", IEEE International Conference on Robotics and Automation, May 1994

[7] A. Stentz, "The focused D* algorithm for real-time replanning", International Joint Conference on Artificial Intelligence, August 1995

[8] J.F. Canny and M. C. Lin, "An opportunistic global path planner", Algorithmica, vol. 10, pp. 102-120, 1993

[9] M. Lindhe, P. Ogren, and K.H. Johansson, "Flocking with obstacle avoidance: a new distributed coordination algorithm based on Voronoi partitions", IEEE Conference on Robotics and Automation, April 26-May 1, 2004

[10] R. Siegwart and I.R. Nourbaksh, *Introduction to Autonomous Mobile Robots*, The MIT Press, Cambridge, Massachusetts, 2004

[11] S.S. Epp, *Discrete Mathematics with Applications*, 3rd ed., Brooks Cole, Boston, MA, 2003

[12] K. Hwang, *Advanced Computer Architecture: Parallelism, Scalability, Programmability*, McGraw-Hill, Boston, MA, 1992

[13] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance", IEEE Robotics & Automation Magazine, Vol. 4, pp. 23-33, March 1997

[14] J.J.A.M. Keij, "Obstacle avoidance for wheeled mobile robotic systems (literature exploration)", Technische Universiteit Eindhoven, Eindhoven, The Netherlands, February 17, 2003, Report No. 2003.10

[15] D.I.A.Cohen, *Basic Techniques of Combinatorial Theory*, John Wiley & Sons, Inc., pp. 292, 1978

[16] C. Kavka, M. Schoenauer, "Evolution of Voronoi-based Fuzzy Controllers", International Conference on Parallel Problem Solving From Nature, Birmingham, UK, September 18-22