A Comprehensive Survey
Path finding Techniques for Robotics

Asadullah Shaikh, Terna Engineering College, Nerul, Navi Mumbai

Dr. Vaishali Khairnar, Professor, Terna Engineering College, Nerul, Navi Mumbai

***Abstract*:** Many important applications, such as GPS, video games, robotics, logistics, and crowd simulation, rely on pathfinding, which is the process of finding the best route or path in a given environment. Pathfinding can be done in different ways and for different purposes, depending on the situation. Although there have been many advancements in pathfinding algorithms in the past 20 years, the subject remains a popular area of study. the survey provides an overview of the difficulties encountered in generating graphs, and it discusses the most frequently used pathfinding algorithms and strategies. We assess the influence of recent advancements and enhancements in current methods on the fields of robotics and video games. We have categorized pathfinding algorithms by analyzing them in a 2D/3D environment. this is a boring sentence that tells us what the survey is about. It says that the researchers want to know what they have done in this subject in the last ten years, how they did it, and what they found out.

Keywords : path finding algorithm, GPS, crowd simulation, etc.

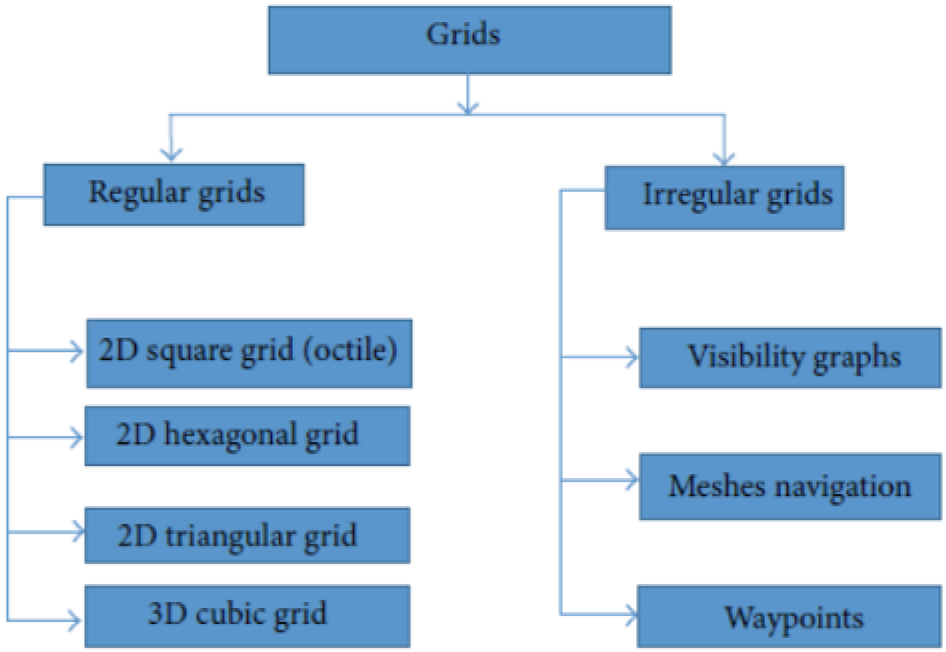
1. INTRODUCTION

Pathfinding is used in GPS [1], video games, robotics, logistics and crowd simulation,[2][3] in static, dynamic or real time settings. Over the last twenty years there have been many improvements made that have increased both accuracy and efficiency in path finding algorithms yet still it remains a topic of great interest. The most crucial area is now considered to be delivering high-performance realistic routes for users. Single-agent pathfinding search, multiagent pathfinding search, adversarial pathfinding, dynamic changes in the environment, varied terrain, mobile units, and insufficient information are all examples of pathfinding problems[4]. Each of these issues has a variety of applications in various fields. Graph creation and a pathfinding algorithm are the two fundamental processes in pathfinding. The "terrain topology" graph generation problem is regarded a cornerstone of robotics and video game applications. Pathfinding navigation is carried out in this problem in a variety of continuous environments, including known 2D/3D environments and unknown 2D environments.

The graphs for these scenarios have been illustrated in various ways to depict the navigation environment. This paper discusses each of the representative environment graphs, which are either related to skeletonization or cell decomposition [5]. Methods for extracting a skeleton from a continuous environment. This skeleton outlines the fundamental structure of the traversable space by establishing a graph G, where V represents a set of vertices that correspond to a continuous environment coordinate, and E represents a set of edges that connect vertices in line-of-sight. Skeletonization methods can lead to two distinct types of irregular grids: a visibility graph or a waypoint graph. The continuous space in an environment is divided into smaller sections using cell decomposition methods. A cell is usually represented by a round or convex shape and signifies a part of the area without any obstructions. the agents can move in a straight line between any two points within the same cell, as the cells are either circles or convex polygons with no obstacles. The following are some beliefs about the characteristics of maps that are made by separating the skeleton and the cells of the terrain.

i. The process of enlarging the terrain maps will provide more room and variety to the initial map that is unnecessary.
ii. The characteristics of maps with a grid pattern (regular and irregular) change a lot depending on whether they are used for games or robots.
iii. When making fake terrain maps for testing robots, the characteristics of real terrain maps should be considered.

After the pathfinding process, the search algorithm is the second step. The goal is to find the most effective way to guide consumers back to their desired destination. The creators of video games and robots use various techniques. A popular search algorithm utilized in gaming and robotics is A [6]. This was the initial algorithm that used a heuristic function to navigate a search network in the optimal order, starting from the source node and expanding until the goal node was found. To evaluate the effectiveness of these algorithms, we need to examine their execution speed, memory usage, and whether the search system operates in a stable, changing, or predictable manner.



(a)

FIGURE 1.1. Types of Grids

The research is an investigation into game pathfinding that considers the future direction. We have examined recent advances in pathfinding and identified some of the problems we face today. The robotics and video gaming industry is a perspective this article offers with the aim of motivating researchers and developers. Figure 1.1 simply illustrates how various types of terrain topologies are connected.

We will look at several grid-based strategies next as well as hierarchical techniques in section three. Agent-based topologies and environments are described in section four, while section five summarises the paper.

Graphs are typically represented by grids wherein vertices or points are linked by edges [7]. The performance of most pathfinding algorithms' navigation is determined by its graph representation attributes. This includes regular grids and irregular grids which we explain extensively on this paper.

2. TYPES OF GRIDS

Regular Grids: Regular grids are one of the most well-known graph forms, and they're popular among game developers and roboticists alike. A number of video game producers have worked in this area, generating titles like Dawn of War 1 and 2, Civilization V, and Company of Heroes; roboticists have used regular grids in the spirit and opportunity of the Mars rovers [8]. Regular grids explain tessellations of regular polygons in 2D and 3D environments (i.e., equilateral and equiangular polygons). The only regular polygons that can be used to tessellate continuous 2D environments (Figures 2.1 (a)–(c)) and 3d Cubic grids (Figure 2.1 (d)) are hexagons, squares, and triangles.

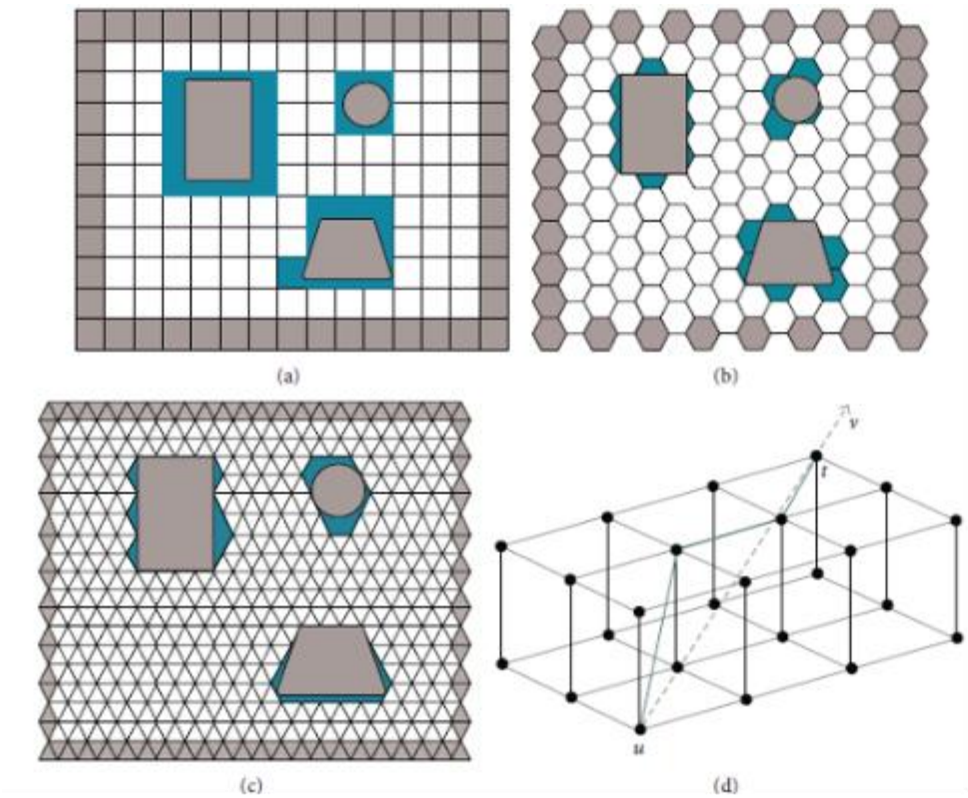


FIGURE 2. 1. Types of Grid Layouts

Irregular Grids: Computational geometry and robotics rely heavily on irregular grids, especially for visibility graphs, navigation meshes, and waypoints. By using irregular grids, visibility graphs can calculate the most direct path for efficient navigation. As navigation meshes, they offer adaptable and precise depictions of navigable regions in intricate surroundings. As markers, they allow the creation of routes by connecting important places in uneven landscapes. These applications demonstrate how irregular grids can improve computational efficiency and adaptability in practical situations.

3. HIERARCHICAL TECHNIQUES

Memory usage is a drawback of employing methods that rely on both regular and irregular grids. By breaking down the continuous environment into smaller parts, hierarchical methods help to reduce the memory space needed. By using smaller details in certain regions, especially around obstacles, a more precise image can be obtained. When intricate details are not necessary, like expansive open spaces, a more coarse texture is employed. The Figure 3.1 shows the techniques used of regular and irregular grids.

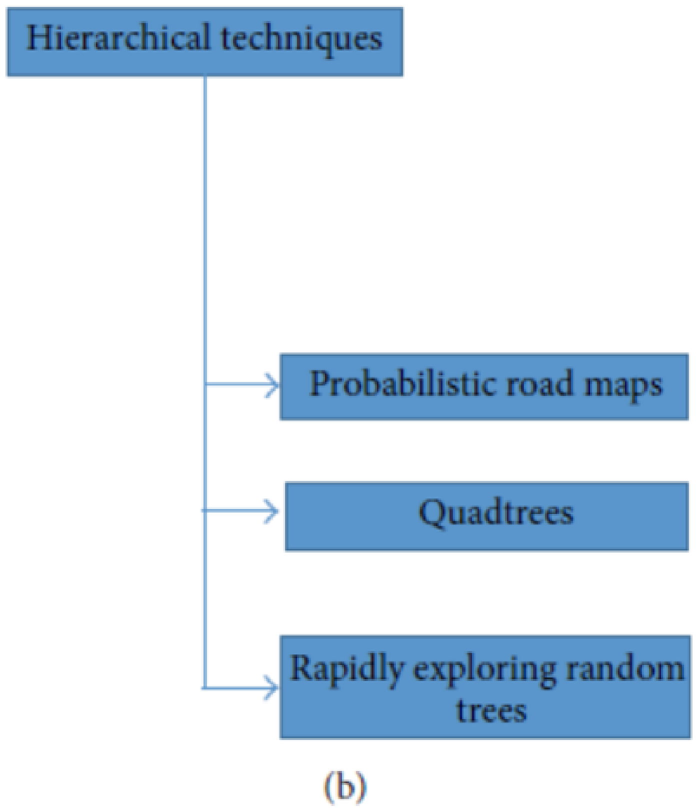


FIGURE 3.1.. Hierarchical Techniques for grid exploration

4. LITERATURE SURVEY

4.1.1. Square Framework in 2 Measurements (Octile):

In computer recreations and mechanical technology, square lattices are one of the foremost common framework charts, and numerous calculations have been recommended to handle the pathfinding issue for this shape of network (see Figure 2(a)). The bounce point look (JPS) single-agent calculation was recommended by Harabor and Grastien [9] to handle a common issue in diversions and mechanical autonomy, particularly the well-known "uniformcost octile lattice in inactive environment." JPS is roughly ten times speedier than A* [6] and contains a moo memory overhead. They evaluated their work utilizing Sturtevant's standard collection of pathfinding maps as a benchmark.

Uras et al. [10] published a strategy for building subgoal charts to speed up way look. The most reason of this ponder is to discover a cell grouping that causes an specialist to continuously go into a basic subgoal. Two-level subgoal charts and two-level subgoal charts with pairwise separations are two varieties of this strategy. There are a most extreme of eight neighbors in a square lattice.

4.1.2 Hexagonal 2D Framework:

Hexagonal networks (Figure 2(b)) contain numerous of the specified qualities of square frameworks, agreeing to Bjornsson et al. [11]. In expansion, hexagonal frameworks take less time to look and have less memory prerequisites than framework charts made of squares. To replicate robot investigation, Quijano and Garrido [12] utilized six hexagonal-grid calculations. They illustrated that the calculations beat quadrangular network methods for both single-agent and multiagent circumstances.

4.1.3 Two-dimensional triangular framework:

Triangular frameworks (Figure 2(c)) are less common than square and hexagonal lattices, but they do have a few points of interest. Demyen and Buro [13] proposed a strategy that employments constrained Delaunay triangulation to diminish look exertion. The authors explored the impact of the environment on development amid a route errand by changing the measure of objects. On enormous maps, their TA and TRA calculations were found to function flawlessly.

4.1.4. Cubic Network in 3D:

The cubic grid (Figure 2(d)) could be a customary chart based on a ceaseless 3D environment, not at all like the lattice charts examined already. Nagy [14] displayed a blended hexagonal and cubic network that calculated non symmetric separations and associated the hexagonal network representation through a cubic network plane. The way finding problem has seen small advancement based on this network chart, but the creator focused that this framework introduction is appropriate to computer illustrations and picture handling.

4.1.5. Graphs Visibility:

The perceivability chart (Figure 3(a)) could be a basic structure in various spaces of geometric chart hypothesis and computational geometry, and it has started intrigued in a assortment of issues [15]. Perceivability charts have been utilized in a assortment of applications, most eminently within the calculation of Euclidean most brief ways within the nearness of obstructions [16]. They moreover used a procedure to quicken the rate of convergence by creating a reasonable introductory populace. Rather than utilizing cell-based strategies like standard networks, the creators utilized the Minkowski whole, a common computational geometry-based methodology. This strategy, in any case, was found to be speedier than past computational geometry approaches, but it gave destitute comes about.

4.1.6. Mesh Navigation:

The route work is spoken to by the walkable segments of a outline, as appeared in Figure 3(b). This work can be spoken to in a variety of ways, counting triangles and polygons. In reality, the work and perceivability charts show up to be indistinguishable, but this is often not the case. Perceivability charts are more complicated than work charts in hone. The larger part of work chart applications are in video diversions. Sisl'ak et al. [17] proposed an A-inspired calculation. Their AA framework, which is based on the single-agent pathfinding issue, permits for flight path planning. The AA calculation was utilized to model nonholonomic plane development in a energetic environment. The capacity to consider a nonzero-sized plane and way arranging for the heading vector goes upwards from the flat plane are two of AA's essential highlights.

5. REVIEW AND DISCUSSION

5.1. Agent-Based Topologies:

Over an assortment of topologies, the single-agent pathfinding look has been broadly inquired about. Researchers' key objective is to deliver an ideal approach in terms of computing time and memory overhead. Besides, a standard network, particularly a square framework (octile), is the finest representation of a plane surface, while an sporadic triangular work is the finest representation of uneven ground. Manufactured maps drawn "algorithmically" for video recreations or robot route have particular qualities than maps drawn by originators. Multiagent pathfinding, unlike single-agent pathfinding, can be decoupled or coupled. Each agent's way is arranged independently within the decoupled strategy. In spite of the fact that such algorithms are greatly speedy, their optimality and even completeness are once in a while ensured. The problem becomes a single-agent look that's unraveled from a higher-dimension point of view within the connected method. One of the foremost troublesome issues is avoiding collisions between operators as they move. In this circumstance, we will oversee the agent's versatility utilizing the Iterative Saddling System (ITF) [18]. ITF can be utilized with the foremost up-to-date single-agent pathfinding method, such as the moved forward JPS [19]. The over calculations have imperfections in that they spend as well much time checking gridmaps and have considerable memory overheads. Numerous analysts neglect the shape of the operators as they travel from their beginning area to their target. This is often particularly true when managing with obstacles' corners and, within the case of multiagent systems, avoiding other operators.

5.2 Agent-Based Environment

The pathfinding comes about can be affected by exploring through various types of situations. The larger part of past work has been exhausted a inactive and real-time setting, with as it were a couple of special cases. Because many areas have pathfinding applications for such scenarios, single-agent pathfinding regularly uses a inactive environment. A number of inquire about have misused multiagent circumstances in real-time and energetic settings. Real-time pathfinding enables the look specialist to require activities whereas the look is ongoing. This implies that operators can get fractional arrangements, permitting them to take after or something else consolidate exercises into the final solution. The creators are especially curious about real-time methodology recreations, which request way arranging from different agents in a shared space whereas following to asset limits and creating high-quality comes about. Table 1 outlines the foremost broadly utilized techniques for both single-agent and multiagent issues.

The Table 5.1 showcases the different attributes compare for various topologies.

TABLE 5.1: Review of multiple topologies and other parameters

Topologies	**Environment System**	**Pathfinding addresses**	**Exemplification**	**Memory complexity**	**Time complexity**	**Cost metric**	**Path finding**	**Reference**
Undirected uniform-cost square grid maps with diagonal movement	Static	Single Agent	Game development	-	A* + ALTBestp $O(n \log n)^{1/7}$ IDA* + ALTBestpfaster	ALTBestp, Manhattan, and ALT heuristics	A* and IDA* algorithms	[20]
Undirected uniform-cost square grid maps without diagonal movement	Static	Single Agent	Game development	-	$O(n \log n)^{1/2}$	Manhattan	Improved A* algorithm	[21]
Undirected uniform-cost square grid maps with diagonal movement	Static	Single Agent	Game development	No memory overhead	JPS algorithm $O(n \log n)^{1/10}$		A*, HP A*, and JPS algorithms	[9]

Undirected uniform-cost square grid maps with diagonal movement	Static	Single Agent	Game development	$O(b^i)$ b = maximum branching factor, i =iteration	-		IEA* and IDA* Algorithms	[22]
Hexagonal grid	Real-time	Multi-agent	Robotics systems	-	SUB algorithm $O(n \log n)^{1/100}$	Euclidean	Augmented A* and Accelerated A* algorithms	[23]
Triangular grid	Real-time	Multi-agent	Robotics and games development	-	-		AD* algorithm	[24]
Cubic Grid	Static	Single Agent	Robotics and games development	-	-		Theta*, Lazy Theta*, and A*	[25]
Mesh navigation	Real-time and dynamic	Multi-agent	Robotics and games Development	45 KB of memory per agent	-		Framework	[26]
Visible graph	Dynamic	Multi-agent	Robotics development	AA* less than A*	-	Euclidean	AA* algorithm	[27]

6. CONCLUSION AND FUTURE TRENDS

We have highlighted current progressions within the field of pathfinding in this audit article. The fundamental classes and approaches that are presently utilized for pathfinding have moreover been altogether investigated. It is obvious from the investigate referenced here that critical endeavors were made to find rectify real-time pathways with moo or no unsettling influence to the test in address. The underlying shortest-path standards in mechanical autonomy and video recreations are clearly youthful speculations, and we should anticipate huge developments in route within the coming years. Path-planning calculations are being progressed by analysts from all around the world. Increased reality is one space that hasn't been looked at however. This point offers a assortment of openings and challenges to scholastics, and we accept that the next-generation video amusement industry will be built on the intelligently possibilities of expanded reality.

REFERENCES

[1] N.R.Sturtevant and R.Geisberger, “A comparison of high-level approaches for speeding up pathfinding,” in *Proceedings of the 6th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE ’10)*, pp. 76–82,October 2010.

[2] B. Bonet and H.Geffner,“Planning as heuristic search,” *Artificial Intelligence*, vol. 129, no. 1-2, pp. 5–33, 2001.

[3] M.Helmert,*Understanding PlanningTasks: Domain Complexity and Heuristic Decomposition*, vol. 4929 of *Lecture Notes in Computer Science*, Springer, Berlin, Germany, 2008.

[4] A. Botea, B.Bouzy, M.Buro, C.Bauckhage, and D. S. Nau, "Pathfinding in games," in *Artificial and Computational Intelligence in Games*, pp. 21–31, 2013.

[5] H.M. Choset, *Principles of Robot Motion: Theory, Algorithms, and Implementation*, MIT Press, Boston, Mass, USA, 2005.

[6] P.E. Hart, N.J.Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," IEEE Transactions on Systems Science and Cybernetics, vol. 4, no. 2, pp. 100–107, 1968

[7] T.Ma, Q.Yan,W. Liu, D. Guan, and S. Lee, "Grid task scheduling: algorithmreview," *IETE Technical Review*, vol. 28, no. 2, pp. 158–167, 2011.

[8] J. Carsten, A.Rankin, D. Ferguson, and A. Stentz, "Global planning on the mars exploration rovers: software integration and surface testing," *Journal of Field Robotics*, vol. 26, no. 4, pp. 337–357, 2009

[9] D.D. Harabor and A. Grastien, "Online graph pruning for pathfinding on grid maps," in Proceedings of the 25th National Conference on Artificial Intelligence (AAAI '11), San Francisco, Calif, USA, 2011

[10] T.Uras, S.Koenig, and C. Hern´andez, "Subgoal graphs for optimal pathfinding in eight-neighbor grids," in *Proceedings of the 23rd International Conference on Automated Planning and Scheduling (ICAPS '13)*, Rome, Italy, June 2013.

[11] Y. Bj¨ornsson,M. Enzenberger, R. Holte, J.Schaeffer, and P. Yap, "Comparison of different grid abstractions for pathfinding on maps," in *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI '03)*, pp. 1511–1512, Acapulco,Mexico, August 2003.

[12] H. J. Quijano and L.Garrido, "Improving cooperative robot exploration using an hexagonal worldrepresentation," in *Proceedings of the Electronics, Robotics and Automotive Mechanics Conference (CERMA '07)*, pp. 450–455, September 2007.

[13] D. Demyen and M.Buro, "Efficient triangulation-based pathfinding," in *Proceedings of the 21st National Conference on Artificial Intelligence and the 18th Innovative Applications of Artificial Intelligence Conference*, pp. 942–947, July 2006.

[14] B. Nagy, "Calculating distance with neighborhood sequences in the hexagonal grid," in *Combinatorial Image Analysis*, pp. 98–109, Springer, 2005.

[15] S. K. Ghosh, *Visibility Algorithms in the Plane*, Cambridge University Press, 2007.

[16] T. Lozano-P´erez and M. A.Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, 1979.

[17] D. ˇSiˇsl´ak, P. Volf, and M. Pechoucek, "Flight trajectory path planning," in *Proceedings of the Scheduling and Planning ApplicationsWorkshop*, pp. 76–83, 2009.

[18] Z. Bnaya, R. Stern, A. Felner, R. Zivan, and S. Okamoto, "Multiagent path finding for self interested agents," in *Proceedings of the 6th Annual Symposium on Combinatorial Search*, pp. 38–46, July 2013.

[19] D. Harabor and A. Grastien, "Improving jump point search," in *Proceedings of the 24th International Conference on Automated Planning and Scheduling*, 2014.

[20] T. Cazenave, "Optimizations of data structures, heuristics and algorithms for path-finding onmaps," in *Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG '06)*, pp. 27–33, Reno,Nev,USA,May 2006.

[21] D. Harabor and A. Botea, "Breaking path symmetries on 4- connected grid maps," in *Proceedings of the 6th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE '10)*, pp. 33–38, October 2010.

[22] C. M. Potts and K. D. Krebsbach, "Iterative-expansion A," in *Proceedings of the 25th Florida Artificial Intelligence Research Society Conference (FLAIRS '12)*, Marco Island, Fla, USA, May 2012.

[23] L. Chrpa and P. Nov´ak, "Dynamic trajectory replanning for unmanned aircrafts supporting tactical missions in urban environments," in *Holonic and Multi-Agent Systems for Manufacturing*, pp. 256–265, Springer, 2011.

[24] M.Kapadia, K.Ninomiya,A. Shoulson, F. Garcia, andN. Badler, "Constraint-aware navigation in dynamic environments," in *Proceedings of theMotion onGames (MIG '13)*, pp. 111–120, 2013

[25] A.Nash and S.Koenig, "Any-angle path planning," *AI Magazine*, vol. 34, no. 4, p. 9, 2013.

[26] M. Kallmann, "Navigation queries from triangular meshes," in *Motion in Games*, vol. 6459 of *Lecture Notes in Computer Science*, pp. 230–241, Springer, Berlin, Germany, 2010.

[27] D.ˇSiˇsl´ak, P.Volf, andM. Pechoucek, "Accelerated A∗ trajectory planning: Grid-based path planning comparison," in *Proceedings of the 19th International Conference on Automated Planning & Scheduling (ICAPS '09)*, pp. 74–81, 2009.