Hindawi Publishing Corporation
International Journal of Computer Games Technology
Volume 2015, Article ID 736138, 11 pages
http://dx.doi.org/10.1155/2015/736138

Review Article

A Comprehensive Study on Pathfinding Techniques for Robotics and Video Games

Zeyad Abd Algfoor, Mohd Shahrizal Sunar, and Hoshang Kolivand

Media and Games Innovation Centre of Excellence (MaGIC-X) UTM-IRDA Digital Media Centre, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia

Correspondence should be addressed to Mohd Shahrizal Sunar; shahrizal@utm.my

Received 30 September 2014; Accepted 18 March 2015

Academic Editor: Yiyu Cai

This survey provides an overview of popular pathfinding algorithms and techniques based on graph generation problems. We focus on recent developments and improvements in existing techniques and examine their impact on robotics and the video games industry. We have categorized pathfinding algorithms based on a 2D/3D environment search. The aim of this paper is to provide researchers with a thorough background on the progress made in the last 10 years in this field, summarize the principal techniques, and describe their results. We also give our expectations for future trends in this field and discuss the possibility of using pathfinding techniques in more extensive areas.

1. Introduction

Pathfinding is a fundamental component of many important applications in the fields of GPS [1], video games [2], robotics [3], logistics, and crowd simulation [4, 5] and can be implemented in static, dynamic, and real-time environments. Although a number of developments have improved the accuracy and efficiency of pathfinding techniques over the past two decades, the problem still attracts a great deal of research. Currently, the most important area concerns the provision of high-performance, realistic paths for users. In general, there are different variations of the pathfinding problem [6, 7], such as single-agent pathfinding search, multiagent pathfinding search, adversarial pathfinding, dynamic changes in the environment, heterogeneous terrain, mobile units, and incomplete information. Each of these problems has different applications in different fields. Generally, pathfinding consists of two main steps: graph generation and a pathfinding algorithm.

The graph generation problem for "terrain topology" is considered a foundation of robotics and video games applications. In this problem, the pathfinding navigation is conducted in different continuous environments, such as known 2D/3D environments and unknown 2D environments. Several different techniques have been proposed to represent the navigation environment for the graphs of these three scenarios. Each of the representative environment graphs in this paper refers to one of two techniques, skeletonization or cell decomposition [8, 9].

Skeletonization techniques extract a skeleton from the continuous environment. This skeleton captures the salient topology of the traversable space by defining a graph $G = (V, E)$, where V is a set of vertices that map to a coordinate in the continuous environment and E is the set of edges connecting vertices that are in the line-of-sight of one another. Generally, skeletonization techniques may produce two types of irregular grid, namely, a visibility graph or a waypoint graph. Cell decomposition techniques decompose the traversable space in the continuous environment into cells. Each cell is typically defined by a circle or convex polygon and represents a region of traversable space with no obstacles. Because the cells are circles or convex polygons that do not contain obstacles, agents can travel in a straight line between any two coordinates within the same cell (without path planning).

There are a number of hypotheses about the properties of the terrain maps produced by skeletonization and cell decomposition as follows.

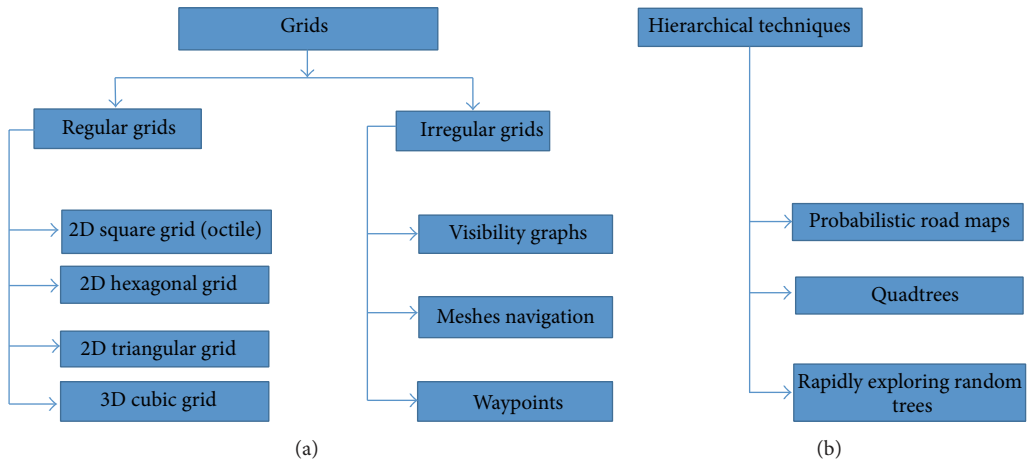

Figure 1: From left to right: (a) terrain topologies; (b) hierarchical techniques.

(i) Scaling the terrain maps will create unnecessary space and variation in the original map.

(ii) There is a significant difference between the properties of grid terrain maps (regular and irregular) depending on whether they are for games or robotics.

(iii) The design of artificial terrain maps for robotics testing must consider the properties of actual terrain maps.

The second step in the pathfinding process is the search algorithm itself. Here, the problem is to return the optimal path to users in an efficient manner. Games and robotics developers use a variety of techniques. A* [10] is one of the best-known search algorithms for games and robotics. It was the first algorithm to use a heuristic function to travel a search graph in a best first manner, the search growing from the start node until the goal node is found. A* inspired many modified and improved algorithms. To evaluate the efficiency of such algorithms, we must take into account the execution time, memory overhead, and whether the environment of the search system is static, dynamic, or real-time "deterministic."

This paper looks at pathfinding research in the field of video games, with an emphasis on future trends and developments. We summarize recent progress in pathfinding and identify current challenges. Our intention is to inspire researchers and developers and to provide an outlook on both the robotics and video games industries. The relationship between different types of terrain topologies is illustrated in Figure 1. In the next section, we introduce various grid-based methods, before describing hierarchical techniques in Section 3. Section 4 presents a discussion on agent-based topologies and environments, and Section 5 summarizes the paper.

2. Grids

A grid is composed of vertices or points that are connected by edges to represent a graph [11]. In most pathfinding algorithms, the navigation performance is based on the attributes of this graph representation. We explain the fundamental concept of two popular grid-based approaches: regular grids and irregular grids.

2.1. Regular Grids. Regular grids are one of the most well-known graph types and are widely used by computer games developers and roboticists. There are a number of video game developers who have worked in this area, producing games such as Dawn of War 1 and 2, Civilization V, and Company of Heroes; roboticists have employed regular grids in the mars rovers spirit and opportunity [12]. In 2D and 3D environments, regular grids describe tessellations of regular polygons (i.e., equilateral and equiangular polygons). Hexagons, squares, and triangles are the only regular polygons that can be used to tessellate continuous 2D environments (Figures 2(a)–2(c)) and 3D cubic grids (Figure 2(d)).

2.1.1. 2D Square Grid (Octile). Square grids are one of the most popular grid graphs in computer games and robotics, and numerous algorithms have been proposed to solve the pathfinding problem for this type of grid (see Figure 2(a)). Harabor and Grastien [13] proposed the jump point search (JPS) single-agent algorithm to solve a common problem in games and robotics, namely, the well-known "uniform-cost octile grid in static environment." JPS is around 10 times faster than A* [10] and has a small memory overhead. They evaluated their work based on the standard dataset of pathfinding maps proposed by Sturtevant [14].

Uras et al. [15] published a method to accelerate the path search by generating subgoal graphs. The main idea of this paper is to identify a sequence of cells such that an agent always moves into a simple subgoal. Variations of this method use two-level subgoal graphs and two-level subgoal graphs with pairwise distances. In a square grid, there is a limit of eight neighbors.

Harabor and Grastien [16] improved the JPS algorithm by employing three strategies. First, they considered nodes as block-based operations and then utilized a system of offline preprocessing. The third strategy is an improved set of current pruning rules. The resulting algorithms were compared with that reported by Uras et al. [15].

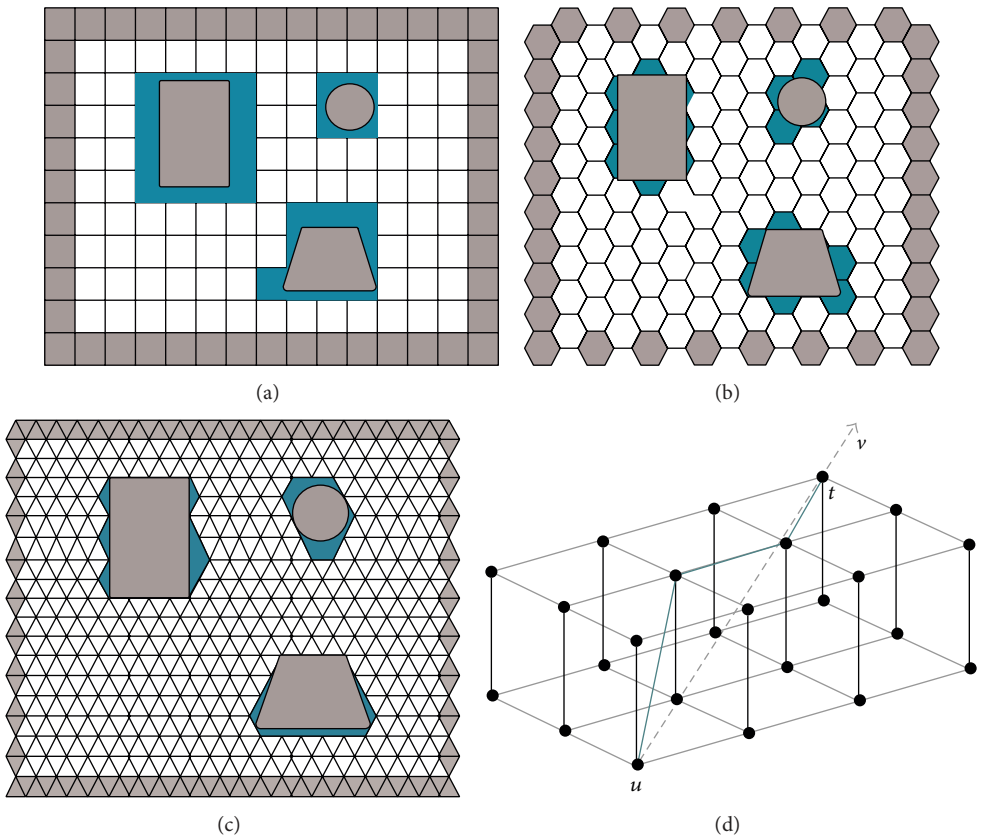

Figure 2: (a) Square grid with three obstacles, (b) hexagonal grid with three obstacles, and (c) triangulation grid with three obstacles and (d) without obstacles.

Bnaya et al. [17] proposed the Exhaustive and Monte-Carlo Iterative Taxing Algorithms (EITA/MC-ITA) for the multiagent pathfinding problem. These algorithms assume agents can detect conflicts during navigation and assign one agent priority to use the path while the other has a penalty "time-stop" added to its total cost. The main drawback of these techniques is that priority is randomly assigned to one or other agent.

In terms of changing graphs in the presence of "dynamic obstacles," Anderson [18] presented an excellent additive heuristic for a single-agent search on a four-connected square grid. This technique effectively reduces the search time in implicit graphs, and the only drawback is that they implemented it with the A* algorithm, which unnecessarily expands the number of nodes during the search. For the multiagent problem, Jin et al. [19] employed the A* algorithm in an XNA game to solve the pathfinding problem. They used the A* algorithm to handle dynamic obstacles in a 2D square grid map, despite the implementation being in a 3D environment. The disadvantage of their work is that A* is not always optimal in static grid maps.

Sharon et al. [20] proposed five online pruning algorithms in their Real-Time Agent-Centered Search (RTACS), which is applicable to the single-agent problem. These algorithms have been used in domains that are unknown a priori. Koenig and Sun [21] developed two different heuristic techniques to speed up the A* algorithm in a multiagent competitive real-time heuristic search. They demonstrated the advantages and disadvantages of real-time and incremental heuristic searches in square grids.

2.1.2. 2D Hexagonal Grid. Björnsson et al. [22] showed that hexagonal grids (Figure 2(b)) have many of the desirable properties of square grids. In addition, hexagonal grids have smaller search time and memory complexities than grid graphs constructed from squares. Quijano and Garrido [23] used six hexagonal-grid algorithms to simulate robot exploration. They proved that the algorithms outperform those on quadrangular grids for both single- and multiagent problems. Chrpa and Komenda [24] proposed a model for smoothing the trajectory of a helicopter in a hex-grid air traffic control planner. They suggested investigating the problem of multiagent collision avoidance and extended their model to include multiagent pathfinding [25].

Othman et al. [26] discussed the problems faced by researchers who use pathfinding algorithms in robotics. One of the main challenges in mobile robotics is the avoidance of real objects. The D* algorithm has been applied to 2D map modelling methods using square tiles, hexagons, enhanced hexagons, and octiles [27]. The results indicate that octiles and enhanced hexagons produce better paths but require more time during the algorithm search in large maps. However, acceptable results are achieved for small and medium size maps. Pathfinding algorithms have mainly been

implemented on square grids, and the multiagent problem on a hexagonal grid has not yet been investigated in dynamic or real-time environments.

2.1.3. 2D Triangular Grid. Triangular grids (Figure 2(c)) are not as widely used as square and hexagonal grids, but they do have certain desirable properties. Demyen and Buro [28] presented a method that reduces the search effort using constrained Delaunay triangulation. By varying the sizes of objects, the authors examined the effect of the environment on movement during a navigation task. Their TA* and TRA* algorithms were found to work perfectly on large maps. Kapadia et al. [29] investigated the impact of the objects in an environment on the agent navigation layer and presented a framework that can handle hybrid graph constraints imposed on the path.

Nagy [30] derived a novel coordinate system consisting of both hexagonal and triangular grid graphs. This new grid can be used for different applications in domains such as computer graphics and image processing. From our perspective, we believe this new topology could be used with deterministic pathfinding methods, such as A* and its variants.

2.1.4. 3D Cubic Grid. Unlike the grid graphs discussed above, the cubic grid (Figure 2(d)) is a regular graph based on a continuous 3D environment [31]. Nagy [32] reported a mixed hexagonal and cubic grid that calculated the nonsymmetric distances and connected the hexagonal grid representation through a cubic grid plane. Little work based on this grid graph has been applied to the pathfinding problem. However, the author emphasized that this grid presentation is applicable to computer graphics and image processing.

Kiliç and Yalcin [33] used a simple algorithm to compute the motion of three types of waves in a 3D environment and used this to solve a robot pathfinding problem. The authors improved the algorithm proposed by Yeniçeri and Yalçin [34] to perform in a 3D environment. In their proposed model, a robot has just six neighborhood cells that can be used to navigate to the goal cell. This is a clear limitation of this model, and the number of neighborhood cells should be increased to include all 26 neighboring cells to improve the resulting path.

Recently, Nash and Koenig [35] published an excellent paper titled "Any-Angle Path Planning." They introduced three different algorithms, A*, Theta*, and Lazy Theta*, for square, hexagonal, triangular, and cubic grids. The aim of these algorithms is to find the smoothest real path through any terrain topology without considering the size of the agent or the shape of the obstacles. Based on the observed performance, they determined shortest paths rather than shortest grid paths. Unfortunately, any-angle path-planning algorithms consume more time than normal pathfinding algorithms.

2.2. Irregular Grids. Irregular grids are used in many different applications and fields. In this survey, we highlight all of the well-known techniques that have been proposed to represent terrain topology.

2.2.1. Visibility Graphs. The visibility graph (Figure 3(a)) is considered a fundamental structure in different fields of geometric graph theory and computational geometry and has attracted research into various challenges [36]. Visibility graphs have been implemented in different applications and have recently been employed in the computation of Euclidean shortest paths in the presence of obstacles [37, 38]. For an excellent survey of visibility graphs, see Ghosh and Goswami [39].

To solve the single-agent problem, Naderan-Tahan and Manzuri-Shalmani [40] proposed a novel genetic algorithm to find an efficient path through a series of dilating obstacles. In addition, they used a strategy to speed up the convergence rate by generating a suitable initial population. The authors used the common computational geometry-based approach of the Minkowski sum, instead of cell-based methods such as regular grids. However, this method was shown to be faster than other computational geometry approaches but produced suboptimal solutions.

For the multiagent problem, Šišlák et al. [41] solved the collision avoidance problem by designing a height performance path-planning algorithm. The Accelerated A* (AA*) proposed by Šišlák et al. [42] has been compared with Theta* [43], rapidly exploring random trees-based planners [44] and the original A* in the search for the shortest any-angle path.

2.2.2. Mesh Navigation. As illustrated in Figure 3(b), the walkable areas of a map represent the navigation mesh. This mesh can be represented in different ways, such as by triangles or polygons. In fact, the mesh graphs and visibility graphs look very similar, but this is not actually the case. In practical terms, visibility graphs are more complex than mesh graphs. The majority of applications for mesh graphs are in video games.

Šišlák et al. [42] introduced an algorithm inspired by A*. Their AA* algorithm enables flight-path-planning based on the single-agent pathfinding problem. The AA* algorithm has been utilized in a dynamic environment to model the nonholonomic airplane movement. The main advantages of AA* are its ability to consider a nonzero-sized airplane and path-planning for the direction vector heads upwards from the horizontal plane. Kallmann [45] investigated the navigation task for several different mesh triangulations. The author showed that triangular meshes can enable the efficient computation of several navigation procedures in the single-agent case.

In multiagent pathfinding, Kapadia et al. [46] proposed a real-time framework for multiagent navigation that uses multiple heterogeneous problem domains for large, complex, and dynamic virtual environments. The domains described in this paper represent popular solutions that are used in both academia and industry. Furthermore, different domains can be connected together.

Harabor and Botea [47] presented a hierarchical annotated extension of the A* algorithm (HAA*) that can be employed in a real-time environment. This algorithm can work with a single abstract graph and can be used to plan a path for agents with heterogeneous sizes and terrain

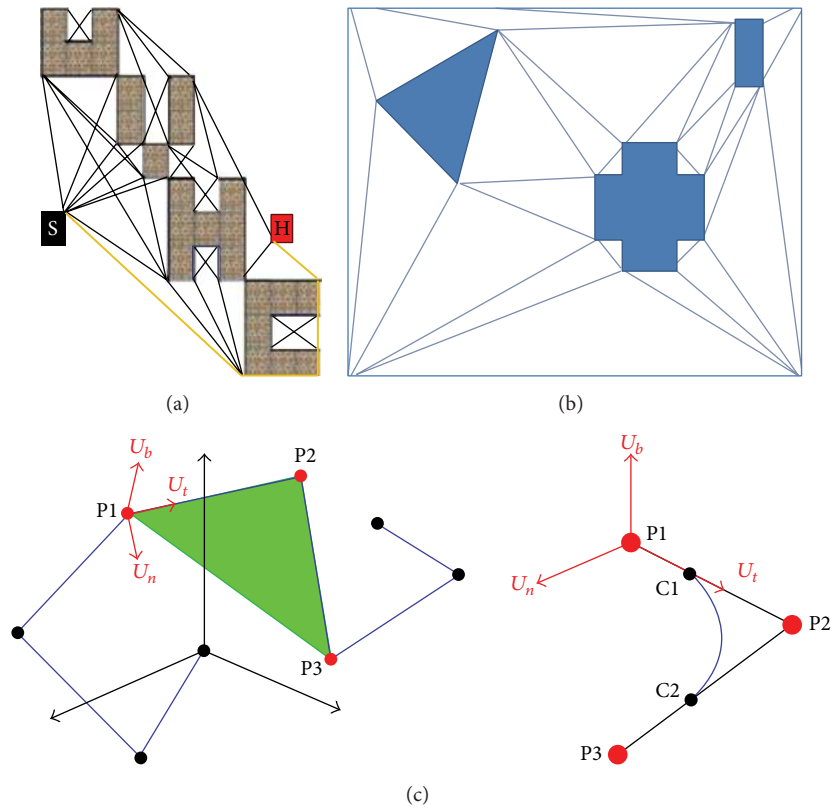

FIGURE 3: (a) Visibility graph (brown represents obstacles), (b) mesh navigation with triangular grid (blue regions represent obstacles), (c) 3D G2CBS path smoothing: three consecutive points in space from a unique 2D plane (left); planar G2CBS path smoothing is applied to these consecutive points after mapping 3D waypoints to 2D (right) [49].

traversal capabilities. The disadvantage of this technique is that, in practice, it produces suboptimal paths for a wide range of problems. However, it does exponentially reduce the computational effort required for low-level searches.

Unlike Kallmann [45], the efficient path-searching system proposed by Chen et al. [48] can be used to implement real-time 3D game scenes with polygons and meshes. Their system converts scene data into an abstract contour model and applies a heuristic search. Sturtevant and Geisberger [1] discussed the complexity of pathfinding in large graphs, such as in the Dragon Age video game. They describe the pathfinding problems faced in the Dragon Age game and conclude that contraction hierarchies and abstractions have a similar overhead and performance and are both useful high-level planning approaches.

2.2.3. Waypoints. Waypoints (Figure 3(c)) are widely used in video games and robotics. Niederberger et al. [49] proposed an A*-based algorithm for pathfinding in static terrains with polygonal obstacles. The algorithm generates a path according to the minimum number of waypoints but suffers from a high computation time and memory overhead.

Ferguson and Stentz [50] developed the Field D* algorithm for interpolation-based planning and described a replanning algorithm for generating low-cost paths through uniform and nonuniform resolution grids. There are two disadvantages to grid-based pathfinding. First, grid-based planners have a limited ability to find the optimal path, because the path must pass via adjacent grid points. Second, the memory requirements of grid-based path planning are often unacceptably high. Burchardt and Salomon [51] proposed an adjusted pathfinding genetic algorithm based on RoboCup's small-size robots. Because of the nature of this dynamic environment, the robot continues to calculate a path until it arrives at the goal.

Yang and Sukkarieh [52] published a paper on "An Analytical Continuous-Curvature Path-Smoothing Algorithm" that fits an ordered sequence of waypoints generated by an obstacle-avoidance path planner. Their results showed that the analytical algorithm can effectively generate a continuous-curvature path that satisfies an upper-bound curvature constraint. The generated path was found to require less effort to track and to minimize control-input variability.

Lucas et al. [53] reported a hybrid algorithm combining constraint solving techniques with ant colony optimization. This method can determine paths in vehicle system architecture problems. In the most complex cases (when mandatory waypoints are distant from the start point and goal point), the method was found to reduce the average computation time by 50% and the preprocessing time by 5%. The authors emphasized that their proposed algorithm is more suitable and could easily solve bigger problems. However, this algorithm cannot find the optimal path faster than deterministic methods.

Cui and Shi [54] published a survey paper on the A*, optimization A*, HPA* [55], and IDA* [56] algorithms and techniques. The aim of this paper was to review the most relevant topics in the academic domain of pathfinding and to encourage games and robotics developers to utilize these algorithms in their industries.

3. Hierarchical Techniques

One disadvantage of techniques that use regular and irregular grids is that they require considerable memory space. Hierarchical techniques mitigate this memory space problem by allowing the continuous environment to be discretized. By applying a finer granularity in certain regions, a more accurate representation can be produced, especially near obstacles. A coarser granularity is applied in regions where detail is not necessary, such as wide, open spaces.

3.1. Probabilistic Road Maps. Coulter [57] proposed a tracking algorithm "pure pursuit algorithm" for robotics in the land based navigation problems. The algorithm calculates the curvature that will move a robot "vehicle" from its current position to target position. The author neglect two significant parameters in his article: dynamic obstacles and natural features like mountains, valleys, streams, and so forth. In the same context, Cai and Goei [58] discussed and developed a mathematical trajectory calculation method for different types of long vehicles. Furthermore, they developed the 3D simulation model for urban city, which includes the turning roads surrounded by buildings and other objects. Choi et al. [59] presented a new pathfinding schema for natural-looking locomotion of a biped figure to facilitate rapid motion prototyping and task-level motion generation. In the single-agent case, the start and goal are defined in a virtual environment, and the proposed schema outputs a sequence of motions to move from the start point to the goal point using a set of live-captured motion clips. This schema consists of three parts: motion generation, roadmap search, and roadmap construction. Unfortunately, this framework is only implemented in the single-agent case and can only handle static obstacles. Dynamic and real-time obstacles make the motion planning more complex.

Kamphuis et al. [60] developed a tactical pathfinding approach for multiagent groups of characters in an urban environment. The path can be obtained in two ways, using an existing road map or extending this to a network of corridors. The authors presented some open problems in this area, such as whether fire crews should split up and perform a so-called bounding over watch procedure and the need for the corridor list to be optimized correctly.

Rohrmuller et al. [61] discussed the safe navigation of a robot through a crowded environment. They proposed an algorithm that uses Markov Chains to model the dynamics of moving agents and predict their future locations. Based on their results, it appears that this approach can successfully find a suitable path in densely populated environments.

3.2. Quadtrees. Finkel and Bentley [62] introduced the Quadtrees data structure. Quadtrees are generated by repeatedly dividing each grid cell in a square grid into four equally sized grid cells. The division process terminates when a grid cell contains no obstacles, or when it reaches some smallest allowable size [63, 64].

Reineking et al. [65] proposed a computational model for representing region hierarchies. The main idea of this algorithm is to limit the search space to the region given by the minimum subtree containing the source and destination region at each step. He et al. [66] presented a novel technique for large scale crowd simulation. Their technique can handle very large crowds and is suitable for dynamically changing environments. A coarse grid is used to represent the macroscopic crowd distribution and motion tendency consistent with fluid dynamics, allowing for a fast implicit update for local path planning and congestion avoidance.

Hirt et al. [67] discussed the efficient use of Quadtree storage for terrain representation and pathfinding. Based on their findings, Quadtree searches seem to be efficient if the map is appropriately represented. Dooms [68] assumed that current pathfinding algorithms are not efficient enough to provide optimal real-time pathfinding in multiagent dynamic environments. He tested the Quad* pathfinding and Quadtree construction algorithms and found that Quad* pathfinding realizes a great improvement in execution times compared to the A* algorithm.

3.3. Rapidly Exploring Random Trees. Naveed et al. [69] developed two Monte-Carlo pathfinding approaches, the Upper Confidence Tree (UCT) and Rapidly Exploring Random Tree (RRT). These were applied to the pathfinding problem in real-time strategy games. The authors showed that UCT outperformed RRT in terms of search effort. However, during gameplay, RRT tended to perform better than UCT. For a survey of Monte-Carlo tree search techniques in the field of video games, see Browne et al. [70].

4. Discussion

4.1. Agent-Based Topologies. The single-agent pathfinding search has been studied intensively over different topologies. The main concern for researchers is to provide an optimal path with respect to computation time and memory overhead. Moreover, the best representation of a plane surface is given by a regular grid, specifically a square grid (octile), and uneven land is best represented by an irregular triangular mesh. Drawing artificial maps "created algorithmically" for applications such as video games or robot navigation have different properties than maps created by designers.

Unlike the single-agent case, multiagent pathfinding can take a decoupled or coupled approach. In the decoupled approach, paths are planned for each agent separately. Such algorithms are very fast, but their optimality, and even completeness, is generally not guaranteed. In the coupled approach, the problem becomes a single-agent search that is solved from a higher-dimension perspective. One of biggest problems is avoiding collisions between agents during their movement. In this case, we can use the Iterative Taxing

Table 1: Recently reported pathfinding algorithms used in robotics and video games.

Topologies	Environment "system"	Pathfinding addresses	Exemplification	Memory complexity	Time complexity	Cost metric	Pathfinding technique	Reference
Undirected uniform-cost square grid maps with diagonal movement	Static	Single-agent	Game development	—	A* + ALTBestp $O(n \log_n)^{1/7}$ IDA* + ALTBestp faster	ALTBestp, Manhattan, and ALT heuristics	A* and IDA* algorithms	[71]
Undirected uniform-cost square grid maps without diagonal movement	Static	Single-agent	Game development	—	$O(n \log_n)^{1/2}$	Manhattan	Improved A* algorithm	[72]
Undirected uniform-cost square grid maps with diagonal movement	Static	Single-agent	Game development	No memory overhead	JPS algorithm $O(n \log_n)^{1/10}$	Manhattan	A*, HPA*, and JPS algorithms	[13]
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	[73]
Undirected uniform-cost square grid maps with diagonal movement	Static	Single-agent	Game development	—	SUB algorithm $O(n \log_n)^{1/100}$	—	SUB, BlockA*, CPD-full, CPD-mbm, JPS-offline, JPS-online, PDH, PPQ, and Tree	[15]
Undirected uniform-cost square grid maps with diagonal movement	Real-time	Multiagent	Game development	—	—	—	ITF, EITA, and MC-ITA schemes	[17]
Undirected uniform-cost square grid maps with diagonal movement	Real-time	Multiagent	Game development	—	PRS algorithm $O(n \log_n)^{1/10}$	Manhattan and Euclidean	A*, FS, PBS, and PRS algorithms	[74]
Hexagonal grid	Real-time	Multiagent	Robotics systems	—	—	Euclidean	Augmented A* and Accelerated A* algorithms	[25]
Hexagonal grid	Real-time	Single-agent	Robotics development	—	—	—	D* algorithm	[26]

Table 1: Continued.

Topologies	Environment "system"	Pathfinding addresses	Exemplification	Memory complexity	Time complexity	Cost metric	Pathfinding technique	Reference
Triangular grid	Real-time	Multiagent	Robotics and games development	—	—	—	AD* algorithm	[29]
Cubic grid	Static	Single-agent	Robotics and games development	—	—	—	Theta*, Lazy Theta*, and A*	[35]
Mesh navigation	Real-time and dynamic	Multiagent	Robotics and games development	45 KB of memory per agent	—	—	Framework	[45]
Visible graph	Dynamic	Multiagent	Robotics development	AA* less than A*	—	Euclidean	AA* algorithm	[41]
Waypoint	Real-time	Multiagent	Robotics development	—	—	—	Ant colony optimization algorithm	[53]

Framework (ITF) [17] to manage the agent's movement. ITF can be hybridized with the most recent rapid technique for single-agent pathfinding, such as in the improved JPS [16].

The shortcomings of the above algorithms are that they spend too much time scanning the grid maps and suffer from high memory overheads. Many researchers do not consider the shape of the agents as they navigate from start point to destination. This is especially true when encountering the corners of obstacles and, in the multiagent case, for avoiding other agents.

4.2. Agent-Based Environment. Navigation through different types of environment can affect the pathfinding results. Most previous work is based on a static and real-time environment, and relatively few works have considered the dynamic case. Single-agent pathfinding generally employs a static environment, because many fields have pathfinding applications for such scenarios. Multiagent cases have been used with real-time and dynamic environments in a few studies. Real-time pathfinding allows the search agent to perform actions while the search is being conducted. This means that partial solutions can be returned, enabling the agent to follow or otherwise incorporate actions into the final solution. The authors are particularly interested in the domain of real-time strategy games, which require path planning for many agents in a common space under the constraints of limited resources and high-quality results. A summary of the most widely used techniques for both single- and multiagent problems is given in Table 1.

5. Summary and Future Trends

In this review article, we have summarized recent progress in the field of pathfinding. Basic classes and techniques, which are now in use for pathfinding, have also been discussed in detail. From the literature cited here, it is clear that remarkable efforts had been made to determine accurate real-time paths with minimal or no disturbance to the sample of concern. It is obvious that the basic shortest-path principles in robotics and video games are mature theories, and in coming years we can expect major changes in navigation. Researchers from all over the world are working to improve path-planning algorithms. One domain that has not been investigated here is augmented reality. This field presents researchers with a range of opportunities and issues, and we believe that the next-generation video games industry will be based on the interactive opportunities offered by augmented reality.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

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] H. Kolivand and M. S. Sunar, "A survey of shadow volume algorithms in computer graphics," *IETE Technical Review*, vol. 30, no. 1, pp. 38–46, 2013.

[3] J. van den Berg, R. Shah, A. Huang, and K. Goldberg, "ANA*: anytime nonparametric A," in *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI '11)*, pp. 105–111, August 2011.

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

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

[6] 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.

[7] R. Graham, H. McCabe, and S. Sheridan, "Pathfinding in computer games," *ITB Journal*, vol. 8, pp. 57–81, 2003.

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

[9] S. Russell and P. Norvig, "A modern approach," in *Artificial Intelligence*, vol. 25, Prentice Hall, Englewood Cliffs, NJ, USA, 1995.

[10] 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.

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

[12] 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.

[13] 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.

[14] N. R. Sturtevant, "Benchmarks for grid-based pathfinding," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 2, pp. 144–148, 2012.

[15] T. Uras, S. Koenig, and C. Hernández, "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.

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

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

[18] K. Anderson, "Additive heuristic for four-connected grid-worlds," in *Proceedings of the 3rd Annual Symposium on Combinatorial Search*, 2010.

[19] H. Jin, W. Wei, and L. Ziyan, "Multi-agent pathfinding system implemented on XNA," in *Proceedings of the 4th International Conference on Computational Intelligence and Communication Networks (CICN '12)*, pp. 651–655, Mathura, India, November 2012.

[20] G. Sharon, N. R. Sturtevant, and A. Felner, "Online detection of dead states in real-time agent-centered search," in *Proceedings*

of the 6th Annual Symposium on Combinatorial Search, pp. 167–174, July 2013.

[21] S. Koenig and X. Sun, "Comparing real-time and incremental heuristic search for real-time situated agents," *Autonomous Agents and Multi-Agent Systems*, vol. 18, no. 3, pp. 313–341, 2009.

[22] Y. Björnsson, 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.

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

[24] L. Chrpa and A. Komenda, "Smoothed hex-grid trajectory planning using helicopter dynamics," in *Proceedings of the 3rd International Conference on Agents and Artificial Intelligence (ICAART '11)*, pp. 629–632, January 2011.

[25] L. Chrpa and P. Novák, "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.

[26] M. F. Othman, M. Samadi, and M. H. Asl, "Simulation of dynamic path planning for real-time vision-base robots," in *Intelligent Robotics Systems: Inspiring the NEXT*, pp. 1–10, Springer, 2013.

[27] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI '95)*, vol. 2, pp. 1652–1659, 1995.

[28] 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.

[29] M. Kapadia, K. Ninomiya, A. Shoulson, F. Garcia, and N. Badler, "Constraint-aware navigation in dynamic environments," in *Proceedings of the Motion on Games (MIG '13)*, pp. 111–120, 2013.

[30] B. Nagy, "Cellular topology and topological coordinate systems on the hexagonal and on the triangular grids," *Annals of Mathematics and Artificial Intelligence*, pp. 1–18, 2014.

[31] J. Carsten, D. Ferguson, and A. Stentz, "3D field D*: improved path planning and replanning in three dimensions," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '06)*, pp. 3381–3386, October 2006.

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

[33] V. Kiliç and M. E. Yalcin, "An active wave computing based path finding approach for 3-D environment," in *Proceedings of the IEEE International Symposium of Circuits and Systems (ISCAS '11)*, pp. 2165–2168, May 2011.

[34] R. Yeniçeri and M. E. Yalçin, "Path planning on cellular nonlinear network using active wave computing technique," in *Bioengineered and Bioinspired Systems IV*, vol. 7365 of *Proceedings of SPIE*, May 2009.

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

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

[37] T. Lozano-Pérez 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.

[38] L. G. Shapiro and R. M. Haralick, "Decomposition of two-dimensional shapes by graph-theoretic clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 1, pp. 10–20, 1979.

[39] S. K. Ghosh and P. P. Goswami, "Unsolved problems in visibility graphs of points, segments, and polygons," *ACM Computing Surveys*, vol. 46, article 22, 2013.

[40] M. Naderan-Tahan and T. Manzuri-Shalmani, "Efficient and safe path planning for a Mobile robot using genetic algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 2091–2097, May 2009.

[41] D. Šišlák, P. Volf, and M. 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.

[42] D. Šišlák, P. Volf, and M. Pechoucek, "Flight trajectory path planning," in *Proceedings of the Scheduling and Planning Applications Workshop*, pp. 76–83, 2009.

[43] A. Nash, K. Daniel, S. Koenig, and A. Felner, "Theta*: any-angle path planning on grids," in *Proceedings of the National Conference on Artificial Intelligence*, p. 1177, 2007.

[44] S. M. LaValle, *Rapidly-Exploring Random Trees: A New Tool for Path Planning*, 1998.

[45] 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.

[46] M. Kapadia, A. Beacco, F. Garcia, V. Reddy, N. Pelechano, and N. I. Badler, "Multi-domain real-time planning in dynamic environments," in *Proceedings of the 12th ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA '13)*, pp. 115–124, July 2013.

[47] D. Harabor and A. Botea, "Hierarchical path planning for multi-size agents in heterogeneous environments," in *Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG '08)*, pp. 258–265, December 2008.

[48] S. Chen, G. Shi, and Y. Liu, "Fast path searching in real time 3D game," in *Proceedings of the WRI Global Congress on Intelligent Systems (GCIS '09)*, pp. 189–194, May 2009.

[49] C. Niederberger, D. Radovic, and M. Gross, "Generic path planning for real-time applications," in *Proceedings of the Computer Graphics International (CGI '04)*, pp. 299–306, IEEE, Crete, Greece, June 2004.

[50] D. Ferguson and A. Stentz, "Using interpolation to improve path planning: the field D* algorithm," *Journal of Field Robotics*, vol. 23, pp. 79–101, 2006.

[51] H. Burchardt and R. Salomon, "Implementation of path planning using genetic algorithms on mobile robots," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 1831–1836, July 2006.

[52] K. Yang and S. Sukkarieh, "An analytical continuous-curvature path-smoothing algorithm," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 561–568, 2010.

[53] F. Lucas, C. Guettier, P. Siarry, A.-M. Milcent, and A. De La Fortelle, "Constrained navigation with mandatory waypoints in uncertain environment," *International Journal of Information Sciences and Computer Engineering*, vol. 1, pp. 75–85, 2010.

[54] X. Cui and H. Shi, "A*-based pathfinding in modern computer games," *International Journal of Computer Science and Network Security*, vol. 11, pp. 125–130, 2011.

[55] A. Botea, M. Müller, and J. Schaeffer, "Near optimal hierarchical path-finding," *Journal of Game Development*, vol. 1, pp. 7–28, 2004.

[56] R. E. Korf, "Optimal path-finding algorithms," in *Search in Artificial Intelligence*, Symbolic Computation, pp. 223–267, Springer, New York, NY, USA, 1988.

[57] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," DTIC Document, 1992.

[58] Y. Cai and S. L. Goei, *Simulations, Serious Games and Their Applications*, Springer, 2014.

[59] M. G. Choi, J. Lee, and S. Y. Shin, "Planning biped locomotion using motion capture data and probabilistic roadmaps," *ACM Transactions on Graphics*, vol. 22, no. 2, pp. 182–203, 2003.

[60] A. Kamphuis, M. Rook, and M. H. Overmars, "Tactical path finding in urban environments," in *Proceedings of the 1st International Workshop on Crowd Simulation*, 2005.

[61] F. Rohrmuller, M. Althoff, D. Wollherr, and M. Buss, "Probabilistic mapping of dynamic obstacles using Markov chains for replanning in dynamic environments," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '08)*, pp. 2504–2510, Nice, France, September 2008.

[62] R. A. Finkel and J. L. Bentley, "Quad trees a data structure for retrieval on composite keys," *Acta Informatica*, vol. 4, no. 1, pp. 1–9, 1974.

[63] H. Samet, "Neighbor finding techniques for images represented by quadtrees," *Computer Graphics and Image Processing*, vol. 18, no. 1, pp. 37–57, 1982.

[64] H. Samet, "An overview of quadtrees, octrees, and related hierarchical data structures," in *Theoretical Foundations of Computer Graphics and CAD*, pp. 51–68, Springer, 1988.

[65] T. Reineking, C. Kohlhagen, and C. Zetzsche, "Efficient wayfinding in hierarchically regionalized spatial environments," in *Spatial Cognition VI. Learning, Reasoning, and Talking about Space*, pp. 56–70, Springer, 2008.

[66] X. He, L. Chen, and Q. Zhu, "A novel method for large crowd flow," in *Transactions on Edutainment VI*, vol. 6758 of *Lecture Notes in Computer Science*, pp. 67–78, Springer, Berlin, Germany, 2011.

[67] J. Hirt, D. Gauggel, J. Hensler, M. Blaich, and O. Bittel, "Using quadtrees for realtime pathfinding in indoor environments," in *Research and Education in Robotics—EUROBOT 2010*, vol. 156 of *Communications in Computer and Information Science*, pp. 72–78, Springer, Berlin, Germany, 2010.

[68] A. Dooms, *Parallel multi-agent path planning in dynamic environments for real-time applications [Ph.D. thesis]*, University Ghent, 2013.

[69] M. Naveed, D. E. Kitchin, and A. Crampton, *Monte-Carlo Planning for Pathfinding in Real-Time Strategy Games*, 2010.

[70] C. B. Browne, E. Powley, D. Whitehouse et al., "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.

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

[72] 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.

[73] 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.

[74] S. Brand and R. Bidarra, "Multi-core scalable and efficient pathfinding with Parallel Ripple Search," *Computer Animation and Virtual Worlds*, vol. 23, no. 2, pp. 73–85, 2012.