Finding a Pathfinder

Bjørn Reese
The Maersk Mc-Kinney Moller
Institute for Production Technology,
University of Southern Denmark
Campusvej 55
DK-5230 Odense M
breese@mip.ou.dk

Bryan Stout
4531 Airlie Way
Annandale, VA 22003
bstout@mindspring.com

Abstract

The success of a given pathfinding technique for a computer game depends on the requirements and the assumptions of the game and the constraints it imposes. This paper presents a classification of the factors that influences the performance of pathfinding techniques. This includes the dynamics of the game, the geometry of the players and the environment, the (un)predictability of movement, kinematic and temporal restrictions, interaction rules, and real-time performance. The purpose of this classification is the help developers identify the complexity of the task before choosing a certain approach.

Introduction

Choosing a pathfinding techniques that performs well is important to the success of artificial intelligence in computer games, because pathfinding is a fundamental building-block for the field of Artificial Intelligence. There are a wide variety of pathfinding techniques, and one technique may excel under certain circumstances, but do badly under others. The success of a given approach depends on the requirements and assumptions of the game. The purpose of this paper is not to offer solutions, but to raise awareness about the most common factors that influences the efficiency of pathfinding techniques. The more factors that the developer intends to incorporate into the game, the more complex the pathfinding becomes.

Path Type

Pathfinding is usually associated with finding the shortest path, but there are other possibilities, which may have to be solved differently. These include finding any path, finding a path with maximum coverage of an area (Emmanuel, Fagegaltier, and Liegéois 1994), finding a path with minimal exposure (Hoff, Howard, and Tseng 1995), and finding the set of paths with the maximum capacity to get as many units as possible to the destination (Ahuja, Magnanti, and Orlin 1993).

Planning

Not every game is suited for planned navigation. Often it is sufficient to use a set of simple behavioral or reactive rules, and rely on those to perform spatial movement, collision avoidance, and other navigational requirements.

On the other hand, not every game can be satisfactory handled by such rules. Using reactive rules to navigate through a maze most often yields exceptionally bad results. Rule-based navigation is restricted by the lack of global knowledge, which results in paths of poor quality, or even getting caught in traps it cannot escape (local minima.)

Sometimes the player has an objective rather than a geometric destination. One example is the Pursuit-Evade game, where the objective of the evader is to stay away from the pursuer. Another example is to maintain visibility with another player. In such cases planned navigation is not advantageous.

Dynamics

Some parts of the environment may be able to change during the game. It is convenient to categorize these according to the origin of the change.

- **Static environment:** Navigation is rather easy because the environment is predictable. There already exist a huge number of solutions to this case (Latombe 1991).
- **Movable players and static obstacles:** The presence of other movable players needs to be handled dynamically, because their future paths are unknown. Although their instantaneous direction and speed are known, their future movement cannot be predicted. Fortunately it is often possible to disregard them while planning a path, and handle them locally as they are encountered while traversing the path.
- **Moving obstacles:** If the obstacles can be relocated then we cannot pre-calculate the entire path. It is computationally expensive to re-plan each time

a discrepancy is detected, so more dynamic schemes, such as D* (Stentz 1995) or Kinetic Data Structures (Basch, Guibas, and Hershberger 1997), may be required.

- **Appearing/disappearing obstacles:** If the obstacles not only move, but also are able to appear and disappear, the pathfinding gets further complicated because there is no temporal coherency to utilize.
- **Manipulation:** If the player has the ability to alter the environment during movement, planning gets further complicated. This can be moving (Ben-Shahar and Rivlin 1995), destroying, or even constructing new parts of the environment. Construction may be limited to the availability of building material.

Geometry

The physical appearance of the environment plays an important role.

- **Shapes:** Players and obstacles that can be approximated by simple geometric shapes, such as boxes or circles, are much easier and more efficient to handle than arbitrarily shaped entities. Rotation can also increase complexity, unless the shape is circular.
- **Partitioning:** It is common to use graph based search algorithms to find a path. The graph is found by partitioning the environment, which can be done in many ways. The size and connectivity of the resulting graph influences the performance of the search algorithms, so the partitioning method is important. It should not automatically be assumed that a superimposed uniform grid (tiles) will give the best search performance.
- **Concavity:** Many navigation techniques assumes that the player has to circumnavigate obstacles, rather than be guided through them, as is the case with corridors or mazes. Such structures may invalidate certain navigation techniques, such as navigation by traffic rules.
- **Special Structures:** The environment may contain special structures, such as loops, which are troublesome to the space partitioning methods. Approximation of such structures can either lead to jagged motion if the granularity is too fine, or impenetrable areas if the granularity is too coarse. This can necessitate special treatment of these structures, such as using a non-linear roadmap like an equidistant line through the loop which must be followed (Guldner, Utkin, and Bauer 1994).
- **Varying Terrain:** The penetrability the environment may be more nuanced than binary penetrability, ie. either a part is penetrable or it is impenetrable. Some parts may be difficult, but not impossible, to penetrate. The environment may consist of such different terrain, which implies that the unit cost of traversing different parts of the environment differs.

Uncertainty

- **Limited Information:** The amount of information available may be limited – either intentionally restricted to infuse a sense of realism, or unintentionally like in distributed systems. There may be unknown parts of the environment, or previously visited and mapped parts may have changed. The more information available, the more exact the path can be planned.
- **Exploration:** Exploring unknown territory can increase the amount of available information, but deciding how to explore and which parts to explore can be difficult.
- **Occurrences of Events:** Passages may be opened and closed, either periodically or based on some action. Knowledge about such events and their occurrences can be incorporated into the planning.
- **Predictability of Behaviour:** The other players may exhibit a certain type of behaviour. Even though the exact future location of these players cannot be predicted, an approximated or probalistic estimate can be used. Such an estimate may be necessary if a moving target must be intercepted.

Miscellaneous Factors

- **Limited Resources:** The lack of sufficient resources can have a big impact on a given pathfinding technique.
  - **Timely Response:** The quality of the path can be compromised by the available computational resources. Computer games can be divided into turn-based and real-time games. The extreme case which requires real-time response is often present in computer games.
  - **Limited Memory:** Not every pathfinding techniques is able to operate successfully with a limited amount of memory. Some techniques can be altered to work under such conditions, but it usually requires a time-space trade-off. An example is IDA* (Korf 1985) which is a memory bounded version of the A* algorithm.
- **Accuracy:** The level of spatial accuracy of the path is important. Less accuracy will usually cause the planning to be faster and require less memory, and it puts less constraints on local collision avoidance. However, less accuracy can produce inefficient or unnatural movement patterns.
- **Kinematics:** Kinematic constraints, such as gravity, inertia, and limited speed, may have an influence on what types of paths can be used in a solution. The physical coupling of players can also alter the way they have to move.
- **Interaction:** Players may have to interact, ie. cooperate or counteract, to achieve their objectives. An example is two players trying to pass each other in

a corridor that is too narrow. They have to coordinate their movement to prevent blocking each others path. Such deadlocks can either be solved by predetermined traffic rules, or by negotiations between the players. Another example is moving in a formation.

- **Urgency:** There may be explicit or inherent temporal constraints in the game. Objectives can become unimportant if they are not obtained within a certain deadline. For example, bridges may burn, and food may rot.
- **Moving Target:** The destination may move continuously or abruptly during the search, which may require re-planning of the path.
- **Learning:** Building a robust learning technique is a difficult task. It often takes a long time to teach a player, so it is often done during development, and the learned behaviour does not change during the game. Learning in path planning is usually reduced to building a map.

Improving Performance

Changes to the environment can be handled more efficiently if they are anticipated.

Re-planning

If the environment changes or new information becomes available, it may be necessary to re-evaluate the planned path. There are several approaches:

- **Path Re-calculation:** We can discard all previously calculated paths, and start again from scratch with the newly available information. This is robust method, but it is computationally expensive.
- **Path Adjustment:** We can skip the part of the path which passes through the area where the information has been updated. Then we can try to find a new path between the parts of the original path, or the start and the destination if they are closer. This method requires less computational resources, but optimality is no longer guaranteed.
- **Path Granularity:** We can reduce the level of detail of the planned path the further away we are from our current position. Close to our position we make a fine grained plan, and the granularity of the path decreases the further away we get. This way we do not waste too much computational resources on parts which are far away (and more likely to change.)
- **Path Redundancy:** We can expect failures to occur and plan with redundancy, ie. have alternative paths in reserve. This requires more memory and more computational resources when the path is planned.

Dynamic Data Structures

Changes to the environment can be handled more efficiently, if the underlying data structure has been designed to expect such changes. Even then real-time responsivity can be further improved.

- **Sensitivity:** The path only has to be re-planned if the changes will result in a new path. Each part of the path can have an associated sensitivity (Booth and Westbrook 1992), which determine how big the change must be before re-planning.
- **Relaxed Updating:** If the discrepancies introduced by the changes does not lead to unacceptable results, then the workload can be spread out over time, by updating a certain part of the data structure in each time step. Further changes that happens during the relaxed updating are delayed until the effects of the first change have been calculated. This is equivalent to Relaxed Balancing (Larsen 1998).
- **Partial Updating:** The data structure may allow changes while we already are in the process of updating previous changes. This can be combined with relaxed updating.

Conclusion

This paper has articulated various factors that influences the complexity, and hence the performance, of pathfinding. These factors have been catagorized according to issues such as the type of the path, the soundness for planning, the dynamics of the game, the geometry of the environment and the players, the (un)certainty of information, and various other factors.

The classification is not exhaustive, and there is some overlap between certain of the described factors. The overlap was allowed to make a classification that would map more directly unto the problem-domain, rather than an orthogonal classification which was abstracted to obscurity.

References

Ahuja, Ravindra K.; Magnanti, Thomas L.; and Orlin, James B. 1993. *Network Flows. Theory, algorithms, and applications.* Englewood Cliffs, New Jersey: Prentice Hall.

Basch, Julian; Guibas, Leonidas J.; and Hershberger, John, 1997. Data Structures for Mobile Data. In the 8th Symposium on Discrete Algorithms, 747-756. `http://graphics.stanford.edu/~jbasch/publications/kinetic/bgh-dsmd-98.ps.gz`

Ben-Shahar, Ohad and Rivlin, Ehud, 1995. To Push or Not to Push - Part I: On the Rearrangement of Movable Objects by a Mobile Robot. Technical Report, CIS9516, Computer Science Department, Technion, Haifa, Israel. `http://www.cs.technion.ac.il/users/biham/cgi-binw/tr-get.cgi/1995/CIS/CIS9516.ps`

Booth, Heather and Westbrook, Jeffery, 1992. A Linear Algorithm for Analysis of Minimum Spanning and Shortest Path Trees of Planar Graphs. Technical Report, YALE/DCS/TR-763, Department of Computer Science, Yale University, New Haven, Connecticut. `http://www.cs.yale.edu/HTML/YALE/CS/HyPlans/westbrook/sensitivity-planar.ps.Z`

Emmanuel, T.; Fagegaltier, L.; and Liegéois, A. 1994. Motion Planning for a Patrol of Outdoor Mobile Robots. In Proceedings of European Robotics and Intelligent Systems Conference (EURISCON'94), Malaga, Spain, 103-139.

Guldner, Jürgen; Utkin, Vadim I.; Bauer, Rudolf, 1994. On the Navigation of Mobile Robots in Narrow Passages: A General Framework Based on Sliding Mode Theory. In Robot Control 1994 (SYROCO'94): a postprint volume from the IFAC Symposium, Capri, Italy, 59-64. Oxford, UK: Elsevier Science.

Hoff, Bruce; Howard, Michael D.; Tseng, David Y. 1995. Path Planning with Terrain Utilization in ModSAF. In Proceeding of the Fifth Conference on Computer Generated Forces and Behavioral Representation, Orlando, Florida, 255-263.

Korf, Richard E. 1985. Iterative-Deepening A*: An Optimal Admissible Tree Search. In Proceedings of the International Joint Conference on Artificial Intelligence, Los Altos, California, 1034-1036. Morgan Kaufmann.

Larsen, Kim S. 1998. Relaxed Balance Home Page `http://www.imada.ou.dk/~kslarsen/RelBal/`

Latombe, Jean-Claude, 1991. *Robot Motion Planning*. Boston: Kluwer Academic Publishers.

Stentz, Anthony, 1995. The Focussed D* Algorithm for Real-Time Replanning. In Proceedings of the International Joint Conference on Artificial Intelligence. `http://www.frc.ri.cmu.edu/~axs/doc/ijcai95.ps`