Moving Path Planning Forward

Nathan R. Sturtevant

Department of Computer Science
University of Denver
Denver, CO, USA
sturtevant@cs.du.edu

Abstract. Path planning technologies have rapidly improved over the last 10 years, but further integration is needed if pathfinding engines are to drive the behavior of even more realistic characters. This paper briefly outlines existing approaches for path planning and suggests scenarios where integration of information about character's relationships could improve the quality of path planning in games.

1 Introduction

Path planning technologies in the games industry have improved significantly as the understanding of pathfinding technologies has grown. There are standard approaches which can be used to reduce the computational cost of path planning and many tradeoffs between memory usage, time, and optimality are understood.

Taken together, this might lead some to think that pathfinding is 'solved'. But, it is only solved in the sense that every other game technology is solved. The games industry is able to produce games each year that successfully use graphics, sound, music, and many other technologies to create compelling (and not so compelling) interactive experiences. However, just because this can be done, it does not mean that they cannot be done better.

What then would it take to say that pathfinding is 'solved'? It is, perhaps, easier to look at graphics technologies to answer this question. We posit that graphics technologies will be 'solved' when we are able to produce graphics which are indistinguishable from real life at resolutions that allow fully immersive experiences. Furthermore, this should be possible on commodity consumer hardware and in real-time. Work on graphics in games is clearly not yet solved.

Path planning technologies within a game usually fall in the arena of Artificial Intelligence (AI), as path planning path is one of the many systems which are used to create realistic characters. The task of solving AI has usually been associated with the Turing test, but the Turing test is based on natural language conversations, which is not the domain of most games. Following the same idea as solving graphics technologies, we posit that game AI will be solved when we can create characters which are indistinguishable from humans in personality, appearance, and motion. While we might want to create experiences with other types of characters, humans are well-able to judge and evaluate the behavior of other humans, so this seems a suitable bar. (There is no ground truth to measure against in the behavior of aliens.)

This first suggests that solving game AI will require solving the problem of graphics, because the work of an artificial agent will almost always need to be visualized[1]. More importantly, path planning is just one of many technologies needed to solve game AI. Here we arrive at the thesis of this short paper: As barriers to high-quality pathfinding are being lowered, the challenge of path planning is not just building a high-quality pathfinding architecture, but to integrate path planning with all other agent behaviors, so we are not just plotting movement, but a character performance fully suited and unique to the character being built.

We first provide a very brief overview of well-understood path planning technologies, many of which have already been used in shipped games. We then develop several examples of how creating a complete character will require the integration of motion planning and other technologies. We then conclude with some open areas of research needed if AI is, at some point, to be solved.

2 Path Planning Technologies

We begin by quickly providing an overview of how path planning architectures are currently structured.

2.1 Single-Agent Search

Almost all search in games is performed either with uninformed breadth-first search, or more commonly, the A* algorithm [1], which is guided by heuristic estimates. There are many dimensions to understanding A* performance, including data structures and heuristic properties. Most heuristics are consistent, meaning that they obey the triangle inequality, but the growing use of constructed heuristics has led to inconsistent heuristics, which often require special adaptations to A* to achieve good performance [2].

When the state space in a game is too large to allow real-time performance, abstraction and refinement approaches can be used to improve the speed of search at the cost of path quality [3]. These approaches are focused on building a smaller graph such that every path in the smaller graph can be inexpensively refined into a path in the full graph. Abstraction and refinement is well-suited to games with dynamic environments, as an abstraction can often be incrementally updated to represent changes in terrain without significant cost, however this depends on the representation being used.

When the world representation is static, or only changes slightly, the cost of search can also be reduced by building better heuristics. There are many approaches for building heuristics. Storing the all-pairs shortest-path data [4] can reduce the cost of planning to constant time, given that there is space to store this data. This data can be compressed in many different ways [5–7]; the effectiveness of each method depends on the topology of the map. These approaches often maintain optimal paths relative to the representation of the state space being used.

[1] This suggests an interesting challenge problem: Create a compelling artificial agent in a game without the use of graphics. This will, however, probably not require path planning technologies.

Taken together, this suggest a standard approach to solving path planning problems. Given a world representation, the world should be abstracted one or more times until it is possible to achieve real-time performance. Abstraction and heuristics can be combined by building heuristics in abstracted graphs, as the abstracted graphs are smaller, and thus the heuristics require less memory.

This general approach works for many humanoid characters, which can move in restricted spaces, but works less well on objects with highly constrained movement, such as vehicles with large turning radii [8]. Games worlds are authored, so many problematic cases for path planning can be avoided by just creating worlds which are slightly easier to navigate, given the provided constraints.

The cost of moving in a world is often modified slightly given other information in the world. In Dragon Age: Origins, for instance, it is more expensive to walk over traps or other area effects, and it is also more expensive to walk close to other characters. Overall, however, these modifications require some extra work when path planning, because they make the heuristic less accurate.

Games which currently push the state of the art in path planning are beginning to use animation-driven locomotion, where character movement is constrained by the animations available to the character. This approach significantly improves the quality of animation and movement in a game, but also significantly increases the complexity of path planning. In an animation-driven search space, heuristics are less accurate because the world becomes a directed graph. Abstraction and refinement approaches can still work here, as long as there is sufficient space to navigate around obstacles [9], but this is an area where more research could improve theoretical guarantees in constrained areas of search.

The fluid movement allowed by animation-driven locomotion is the first step to enabling even more dynamic pathfinding, such as might be seen by an agent performing parkour, where an agent deftly uses the environment to assist moving even more quickly through the world. This requires careful annotation of the environment to inform agents how to move through it efficiently. This is an area where the state of the art is continuing to improve. One of the biggest barriers to improvement here is the cost of building animations.

2.2 Multi-Agent Search

The problem of planning for multiple agents is significantly more complex than planning for a single agent. There are two general approaches for performing search with multi-agents. The first is to plan each agent independently, and then use local repair methods [10, 11] to perform the tasks of path following and collision avoidance together. In large crowd simulations, agents are often given common goal locations, so they can share a pre-computed heuristic with the distance from every agent to the goal [12]. Work on crowds is continually improving, but more work is required to make crowds more realistic.

Importantly, some work in this area has begun to look at social forces in terms of potential fields [13], something we will elaborate on in the next section.

3 Creating AI Characters

We have glossed over many technical challenges in search with multiple agents and in animation-driven locomotion, but these challenges are currently being addressed by researchers and by those in industry. Something that is being addressed less broadly is the full integration of character performances and path planning. We illustrate this with several examples; there are many more examples that we do not discuss here.

3.1 Approaching a Conversation

Consider the situation of Agent A at a party which wishes to talk to another agent, Agent B. There are several ways the conversation can be approached. If Agent A is extremely polite, it can wait for Agent B to be free before approaching the conversation. If Agent B is likely to want to talk to Agent A as well (e.g. they are long lost friends), Agent A can move within Agent B's line of sight, and then approach the group where Agent B is currently conversing. In this way Agent A can politely enter the conversation or draw Agent B out of the conversation. But, suppose that Agent A and B are bitter enemies. Agent A might then push through a crowd to get to Agent B and pull Agent B out of whatever conversation Agent B is currently engaged in.

Each of these options can be fulfilled by the same request of finding a path from Agent A to Agent B, but the path taken, and the interactions between not only Agent A and B, but also with the rest of the crowd will be determined by the social conventions and relationships at stake. Ignoring the costs induced by the relationships will lead to the same shortest path will always be followed, and the resulting behavior will lack the dynamics proper for realism and fall short of what is needed to solve game AI.

3.2 Finding a Seat in a Room

Consider a new situation where Agent A enters a small lecture hall with two possible entrances to find a seat. First, although we could request a path to a particular seat, it is often the case that any seat would be adequate. Thus, it would be more advisable to select a number of suitable seats and simultaneously search from these seats backwards to the agent, as this provides the highest-quality heuristics. The path from the best seat will reach the agent first, and can then be followed backwards to reach the seat. Going further, the agent might have preferences on seats, including seating near other friendly agents, avoiding the front of the room, or avoiding unfriendly agents. Finally, these preferences extend not only to the chosen seat within the lecture hall, but also to the path taken to this seat. If unfriendly agents are congregating at one door to the lecture hall, the agent may wish to enter via the other door. In some circumstances the agent may being willing to take a quite circuitous route to avoid being seen by unfriendly agents, or to ensure being seen by others that the agent wants to impress.

In these cases we not only need to model the social conventions, but we can utilize the A* algorithm in a more complex manner to make decisions with the best information possible.

3.3 Animation

While not strictly in the domain of movement planning, the selection of animations can play a significant role in the quality of a performance by a character. Given the same path to be followed across a room, there are many styles in which the path could be followed, from a Fred Astaire dance to a slow, hunched-over and limping walk. It should not be hard to see that the selection of animations can be crucial to creating a suitable character performance.

3.4 Barriers to Implementation

Pathfinding engines often expose a simple interface to a game engine: plan a path to a given location. But, this is too simple for many tasks. Thus, an important challenge is cataloguing possible pathfinding requests into the simplest possible API, in a similar way to flocking, which can often be reduced to layered applications of selected following and fleeing behaviors.

The pathfinding layer also needs to know about all of the relationships and related constraints between agents in the game. In many game architectures this simply isn't possible, as the right information cannot be efficiently collected and provided to the pathfinding engine. But, even if an appropriate architecture can be used, these constraints can essentially re-weight the entire map. In the limit, with unbounded cost changes, this is equivalent to the topology itself changing, leading to less accurate heuristics and increased pathfinding costs.

Finally, creating, cataloguing, and selecting proper animations for character behavior is an expensive proposition. There is some hope that procedural animation techniques will reduce this cost, but this too is an open area of research.

4 Conclusions

This paper briefly argues that a suitably powerful path planning architecture is a core component to creating realistic character behavior within a game. If the social interactions between characters are ignored during path planning, movement, and animation selection, the characters will lack the necessary realism to live up to the increasing fidelity of graphics technologies. While some work has been done in this area, it is a rich area for further study, and it is important step in blending the technology of path planning with the art of creating believable and compelling agents.

References

1. Hart, P., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics **4** (1968) 100–107
2. Zahavi, U., Felner, A., Schaeffer, J., Sturtevant, N.R.: Inconsistent heuristics. In: National Conference on Artificial Intelligence (AAAI-07). (2007) 1211–1216
3. Sturtevant, N.: Memory-efficient abstractions for pathfinding. (2007) 31–36
4. Floyd, R.W.: Algorithm 97: Shortest path. Commun. ACM **5**(6) (June 1962) 345–

5. Sturtevant, N., Felner, A., Barer, M., Schaeffer, J., Burch, N.: Memory-based heuristics for explicit state spaces. In: International Joint Conference on Artificial Intelligence (IJCAI-09). (2009) 609–614
6. Botea, A.: Ultra-fast Optimal Pathfinding without Runtime Search. In: Proceedings of the Conference on AI and Interactive Digital Entertainment (AIIDE-11). (2011) 122–127
7. Bast, H., Funke, S., Sanders, P., Schultes, D.: Fast Routing in Road Networks with Transit Nodes. Science **316** (2007) 566
8. Jurney, C.: Postprocessing for high-quality turns. In: AI Programming Wisdom 4. Charles River Media (2008)
9. Sturtevant, N.: Optimizing motion-constrained pathfinding. (2009)
10. Snape, J., van den Berg, J.P., Guy, S.J., Manocha, D.: The hybrid reciprocal velocity obstacle. IEEE Transactions on Robotics **27**(4) (2011) 696–706
11. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. In Stone, M.C., ed.: SIGGRAPH, ACM (1987) 25–34
12. Treuille, A., Cooper, S., Popovic, Z.: Continuum crowds. ACM Trans. Graph. **25**(3) (2006) 1160–1168
13. Gayle, R., Moss, W., Lin, M.C., Manocha, D.: Multi-robot coordination using generalized social potential fields. In: ICRA, IEEE (2009) 106–113