A Hybrid Optimal-Approximate Path Planning Algorithm

David Mould
University of Saskatchewan
Department of Computer Science
e-mail: mould@cs.usask.ca

Michael C. Horsch
University of Saskatchewan
Department of Computer Science
e-mail: horsch@cs.usask.ca

Abstract

Path planning is the problem of finding the lowest-cost path between two endpoints in a weighted graph. An optimal algorithm, such as A*, is guaranteed to return the lowest-cost path. However, the computational expense of A* is high on a class of graphs called terrains*, motivating the development of approximate algorithms such as HTAP (the Hierarchical Terrain representation for Approximate Paths). HTAP has computational cost linear in path length, rather than A*'s quadratic complexity, but does not guarantee the lowest cost path. However, HTAP's overhead means that very short paths are disproportionately costly to find. In this paper, we propose a hybrid algorithm which uses HTAP for long paths and A* for short paths.*

We empirically compare the hybrid algorithm to the HTAP algorithm on the basis of computational cost. The hybrid algorithm has a significant performance advantage over HTAP in the case of very short paths, and is the same as HTAP for long paths. We report results for a number of terrains, giving performance profiles with respect to path length.

Keywords*—path planning; approximation algorithms*

1. Introduction

Path planning is a task often met in areas such as robotics and computer games. An agent travelling across a region must automatically decide what route to take, based on the goal location and on the difficulty of traversing subregions.

A common approach is to superimpose a grid on the region, and model the region with a graph based on the connectivity of the grid. We refer to this kind of graph as a *terrain*, to distinguish it from other kinds of graphs, such as those derived from road maps. Also note that the derived graph should be a weighted graph, where edge weights represent the cost of travelling directly between the connected nodes.

In the context of the graph, the essential problem is to find the lowest-cost path between two specified endpoints. Methods for general graph search are common. Methods for general graph search are common. Because Dijkstra's algorithm is too expensive to use repeatedly for real-time queries, heuristic search using A* is very common [5]. A* is problematic for use in real-time for terrains because the commonly used "air distance" heuristic is a very weak source of information about the cost of paths through a terrain. Variants of A* including real-time A* and iterative deepening A* are also used [1], [2], but are not able to repair the fundamental problem of a weak heuristic. In robotics, potential fields, quad-tree representations of continuous space, wavefront propagation and flood-fill are used [4], in addition to techniques already mentioned. These general methods do not exploit the specific characteristics of terrains, such as their regular connectivity and their tendency to contain regions of similar edge costs. The HTAP algorithm (Hierarchical Terrain representation for Approximate Paths), in contrast, is specifically designed for repeated pathing queries on a static terrain [3]. The HTAP algorithm first computes a representation of the terrain offline, and uses the precomputed information to accelerate the online pathing queries.

The HTAP algorithm empirically has cost $O(n)$ in path length, rather than the more typical $O(n^2)$ cost incurred by A*. In consequence, HTAP's strengths are best shown when n is large – that is, when the paths are long. However, the overhead incurred by HTAP makes it disproportionately costly for short paths, while short paths are precisely those for which the $O(n^2)$ cost of A* is most easily tolerated. Therefore, we suggest a hybrid algorithm, which makes use of A* when the path is estimated to be short, reverting to HTAP if a short path cannot be found.

We implemented our hybrid algorithm and conducted a series of numerical experiments on a variety of maps. We found that for short paths, the hybrid algorithm did indeed perform better than HTAP, while retaining HTAP's fast performance in the case of long paths. Detailed results of the comparison are given in the body of this paper.

The remainder of the paper is organized as follows. We describe both A* and HTAP in greater detail in Section 2; this section also contains details of our hybridization of the two methods. Section 3 describes our experimental procedure, results, and discussion. Finally, Section 4 contains closing remarks and pointers to future directions.

2. Algorithms

Our hybrid algorithm combines A* and HTAP. The unmodified A* uses a heuristic to guide opening actions towards nodes likely to lie on the optimal path. HTAP works by first computing a hierarchical representation of the terrain, and then performing cascades of pathing queries on the resulting hierarchy.

Because the hybrid algorithm often invokes HTAP, it still requires the precomputation of HTAP's pyramid structure.

```
1. Obtain from the user a budget for A$^*$, say $\tau$.
2. Estimate A$^*$'s cost for searching, say $\alpha$.
3. If $\tau > \alpha$, search using A$^*$ until either the budget is expended or a path is found.
4. If no path has been found yet, use HTAP to obtain a path.
5. Return the path found.
```

TABLE I
HYBRID ALGORITHM PSEUDOCODE.

Here, we describe the query resolution process that HTAP employs once the pyramid has been built.

The pyramid is a multiresolution representation of the original graph, with the base of the pyramid being the original terrain and each higher level being a decimated version of the one below. At the top sits a graph sufficiently small that we are willing to search it exhaustively. Each node at the base level has representatives at all higher levels; obviously, many nodes share the same representative. A query is resolved by first finding the representatives of the endpoint nodes at the top pyramid level, and obtaining the best path between them in this small graph; the resulting path is then used to constrain the search space for a new shortest-path query in the graph one level lower in the pyramid. The constrained search space is termed the *corridor*. The constrained pathing queries are repeated, one level lower each time, until a path is found at the pyramid base. This final path is HTAP's reported path between the original endpoints.

Although HTAP works extremely well for long paths, the cascade of path queries and the corridor marking process both contribute to a regrettable overhead which is incurred even on very short paths. Thus, despite HTAP's intent of providing accelerated pathing queries, it can have a greater computational cost than the unaugmented A*.

We therefore propose a hybrid algorithm, which employs A* first, in case a short path can be found quickly, and reverts to HTAP in the case that a short path is not found. Not wishing to burden HTAP with additional overhead in the common case when no short path exists, we only attempt the initial A* search when the air-distance heuristic suggests that the path endpoints are close together in the initial graph. This process is laid out in Table I.

The hybrid algorithm requires a quantified budget for the initial A* portion. We call this budget parameter τ: it is the maximum number of nodes to be searched using A*. If τ nodes are opened and the destination is not reached, the results from the partial A* search are discarded and the search is made using HTAP. The parameter τ also dictates when A* should be employed. If a circle of area $\tau = \pi r^2$ centred on the start location does not contain the destination (where the length unit is the length of an edge in the uniform graph) then it is impossible for A* to find the path with only τ opens. We want not only for it to be possible that A* finds a path, we want it likely that A* will find a path; the chance can be increased by attempting A* only when the budget is large compared to the distance being traversed. In our implementation, we attempt A* when the air-distance between the two nodes is no more than $\frac{\sqrt{\tau}}{2}$.

3. Experiment

We tested the hybrid algorithm on a number of selected terrains. We chiefly used terrains derived from images, given the widespread availability of standard test images; images share some characteristics with terrains making them suitable testing grounds. Images are collections of nodes (pixels) fully connected to a small set of nearby neighbours; terrains derived from images have a wide range of edge costs, but pixel intensities (and the resulting edge cost) are correlated with the intensities of nearby pixels, and such local spatial coherence is a feature of real terrains. Note also that sharp boundaries in terrains can be represented in images, and to the same level of resolution.

Visualizations of some of the terrains appear in Fig. 1. We chose the standard Lena, mandrill, and peppers test images, a standard texture image (a sample from Brodatz D54), an image of an actual terrain, two hand-drawn mazes, and a map consisting of uniform noise. The terrains were derived from the images by associating a node with each pixel, and linking neighbouring nodes with edges whose cost was the average of the intensities of the two pixels (minimum 1).

Each path planning exercise consisted of an optimal-path query with endpoints chosen such that the air distance was at most 60 links (ensuring that there would often be a difference between the HTAP and hybrid results). Each query was resolved by HTAP, by A*, and by the hybrid algorithm. Having identical queries permits us to make direct comparisons between the costs of the three algorithms, where the cost of a query is the number of opened nodes.

We gathered data on pathing profiles for different maps and different values of τ. A portion of a sample pathing profile, showing the HTAP and hybrid profiles, appears in Fig. 2. Not shown is the profile for A*, which exhibits the usual quadratic shape. The HTAP profile can be characterized as linear plus overhead; the existence of the overhead was the original motivation for the hybrid algorithm. The hybrid algorithm's profile shows the lack of overhead for very short paths, eventually followed by the shift to the HTAP algorithm for long paths. The paths intermediate between very short and very long are more costly than the corresponding HTAP path, owing to the expense of an unsuccessful attempt to resolve the path using A*.

However, the size of this unsuccessful intermediate region can be controlled. The parameter τ governs the transition between short and long paths: when τ is small, A* is rarely used, but is almost always successful when deployed; when τ grows larger, the chance of attempting A* rises, but so does the chance that A* will be unsuccessful in finding a path within its budget. Thus, the proper choice of τ is governed by a tradeoff; we investigated the shape of the tradeoff for each of the sample terrains.

Our second experiment consisted of the following. For each terrain, and for values of τ ranging from 0 to 1450, we computed a single data point by taking 5000 individual queries and taking the difference in cost between the computation performed by HTAP and the computation performed by the hybrid algorithm. In all cases, the quality of the path found by the hybrid algorithm was at least as good as that of the path found by HTAP, since the hybrid algorithm either found the optimal path or it reverted to HTAP and reported the same path.

Fig. 3 contains a plot of τ versus total improvement. As mentioned above, improvement is the sum of the differences in cost between all hybrid paths and all HTAP paths. We see the lines starting at 0 (the algorithms are equivalent for $\tau = 0$), and initially rising as τ increases. For most of the maps, the improvement reaches a maximum at approximately $\tau = 800$; this corresponds to the initial period of success when A* is less costly than HTAP's overhead, followed by the increasing cost of the hybrid algorithm as A* is less frequently able to find a path within its budget. The two maze maps do not exhibit optima over the range of τ values shown; rather, they show a steady increase in performance. The different behaviour of the mazes we attribute to the success of the air-distance heuristic for very short paths. Since we limited our pathing queries to nodes nearby in the graph, the cases where the nodes are separated by a wall are comparatively rare. They are less rare in the complex maze (which has more walls), resulting in the slower improvement exhibited by the complex maze compared with the simple maze.

We graphed the non-maze results separately in Fig 4. The increase in τ causes a gradual transition from A* succeeding most of the time to A* failing a significant proportion of the time. The minimum in the graph corresponds to the best performance for the hybrid algorithm, when the increased cost from failed attempts to use A* just balance the decreased cost from the successful attempts.

Table II reports the savings gained by using the hybrid algorithm. The values in the table represent the percentage of nodes opened by the hybrid algorithm compared with the nodes needed by HTAP, in those cases when the hybrid algorithm differed from HTAP.

The extent to which the savings reported in Table II can be realized depends on the particular distribution of path queries in the application. If most pathing queries are long, the hybrid algorithm will usually not differ from HTAP. However, in some applications the majority of pathing decisions are short-range; in such cases, the hybrid algorithm is less costly than HTAP, while also not incurring the quadratic cost of A* on the occasional very long pathing query.

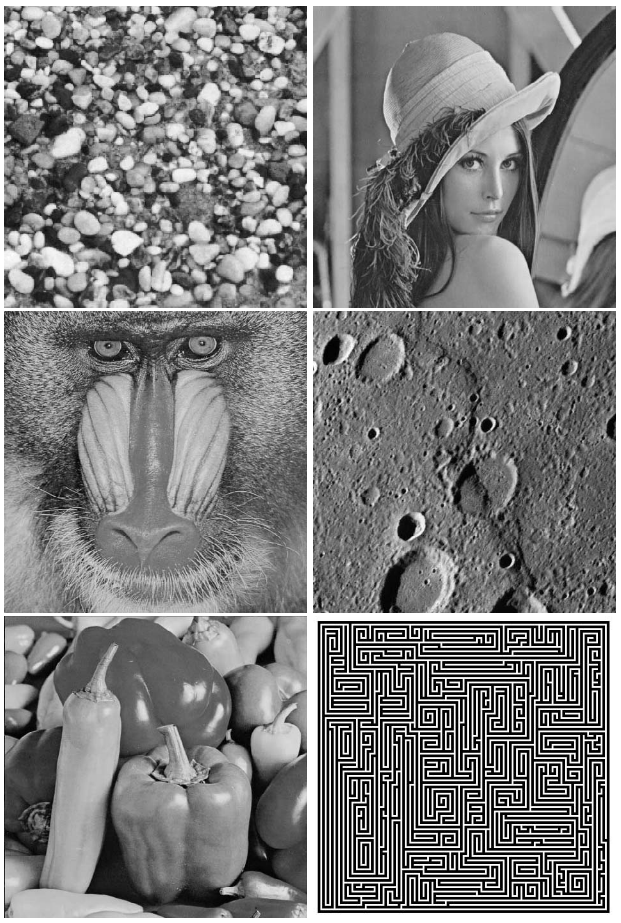
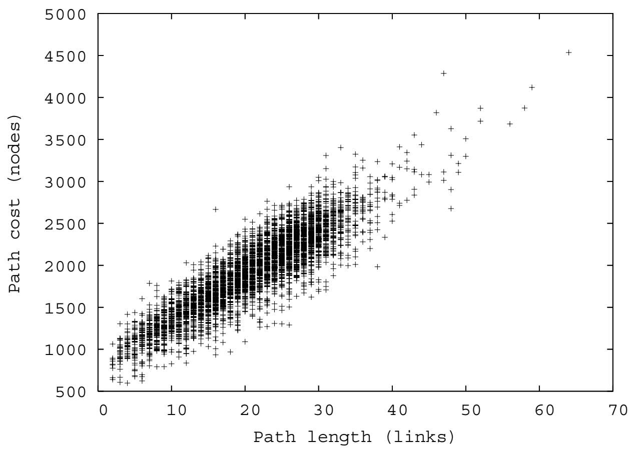


Figure 1. Visualizations of some of the graphs we used. Above, a texture sample and the Lena image; middle, the mandrill image and a terrain image; bottom, the peppers image and a maze.

Map	Hybrid cost (percentage of HTAP)
noise	82.2
complex maze	40.2
simple maze	31.4
Lena	81.2
mandrill	80.5
peppers	84.1
terrain	87.7
texture	88.6

TABLE II

HYBRID SAVINGS SUMMARY.



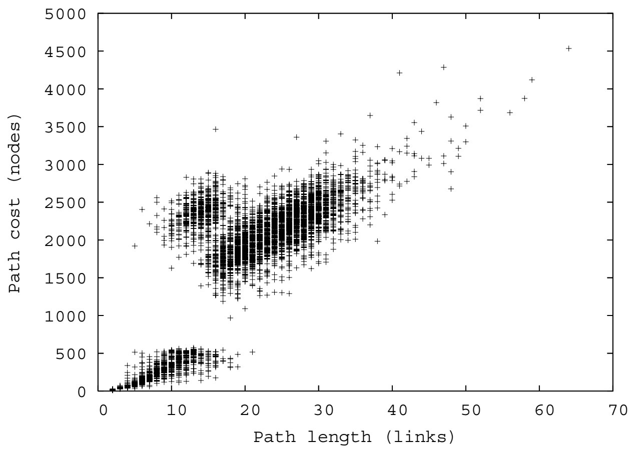


Figure 2. Comparison of the HTAP algorithm to the hybrid algorithm. Above, the hybrid algorithm; below, the hybrid algorithm. For short paths, the hybrid is superior, while for long paths, the two algorithms are identical. There may be a set of intermediate paths where the hybrid algorithm requires more work.

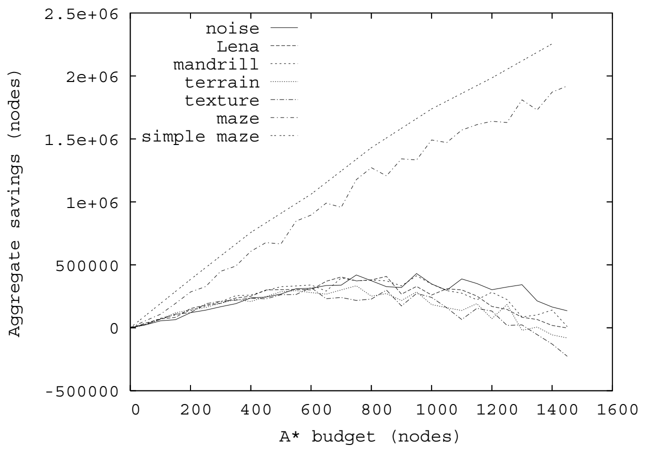


Figure 3: The effect of τ on aggregate cost (all maps).

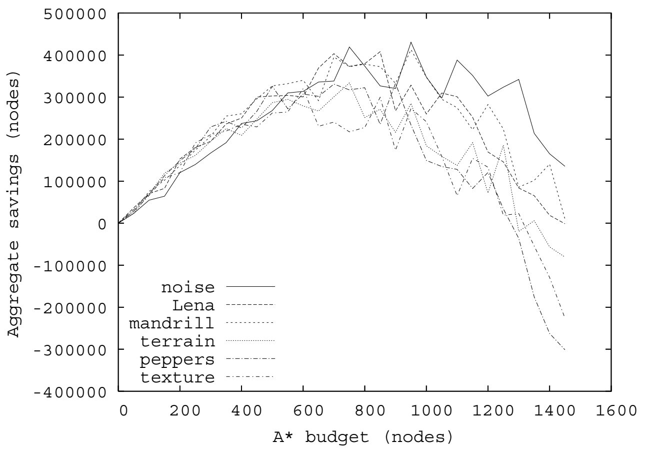


Figure 4. The effect of τ on aggregate cost (mazes removed). There is an optimal value where the hybrid algorithm provides the most benefit.

4. Conclusions

The hybrid algorithm has better performance than either of its two constituent algorithms, given a proper choice of τ. Although it may be difficult to find an optimal value for τ for a particular map, some savings can still be obtained when τ differs from the optimal. In particular, we recommend underestimating τ, since a smaller-than-optimal τ will result in some savings, but a τ larger than the optimal may result in worse performance than HTAP alone.

With a τ at or below the optimum, the hybrid algorithm uses A* for short paths, avoiding the overhead incurred by HTAP, and uses HTAP for intermediate and long paths. The cases where the hybrid algorithm attempts to use A* and fails are more than balanced by the larger number of cases where A* is attempted and succeeds. For long paths, A* is not attempted at all.

Acknowledgments

This work was supported by institutional grants from the University of Saskatchewan and by NSERC grants RGPIN299070-04 and RGPIN238787-01.

References

[1] KORF, R. Iterative-deepening A*: An optimal admissible tree search. In *IJCAI-85* (1985), pp. 1034–1036.

[2] KORF, R. Real-time heuristic search. *Artificial Intelligence 42*, 3 (1990), 189–211.

[3] MOULD, D., AND HORSCH, M. C. An hierarchical terrain representation for approximately shortest paths. In *8th Pacific Rim International Conference on Artificial Intelligence* (2004), pp. 104–113.

[4] MURPHY, R. R. *Introduction to A.I. Robotics*. MIT Press, 2000.

[5] PEARL, J. *Heuristics: Intelligent Search Strategies for Intelligent Problem Solving*. Addison-Wesley, 1984.