A Scouting Strategy for Real-Time Strategy Games

Chen Si
Games Studio
University of Technology, Sydney
chen.si@student.uts.edu.au

Yusuf Pisan
Games Studio
University of Technology, Sydney
yusuf.pisan@gamesstudio.org

Chek Tien Tan
Games Studio
University of Technology, Sydney
chek@gamesstudio.org

ABSTRACT

Real-time strategy (RTS) is a sub-genre of strategy video games. RTS games are more realistic with dynamic and time-constraint game playing, by abandoning the turn-based rule of its ancestors. Playing with and against computer-controlled players is a pervasive phenomenon in RTS games, due to the convenience and the preference of groups of players. Hence, better game-playing agents are able to enhance game-playing experience by acting as smart opponents or collaborators. One-way of improving game-playing agents' performance, in terms of their economic-expansion and tactical battlefield-arrangement aspects, is to understand the game environment. Traditional commercial RTS game-playing agents address this issue by directly accessing game maps and extracting strategic features. Since human players are unable to access the same information, this is a form of "cheating AI", which has been known to negatively affect player experiences. Thus, we develop a scouting mechanism for RTS game-playing agents, in order to enable game units to explore game environments automatically in a realistic fashion. Our research is grounded in prior robotic exploration work by which we present a hierarchical multi-criterion decision-making (MCDM) strategy to address the incomplete information problem in RTS settings.

Categories and Subject Descriptors

I.2.1 [**Applications and Expert Systems**]: Games

General Terms

Algorithms, Experimentation and Performance

Keywords

Artificial Intelligence in Games, Scouting Strategy, Exploration, Real-Time Strategy Games.

1. INTRODUCTION

Real-time strategy games are strategic simulations of battle scenarios. Simulations in RTS environments vary in complexity, and can mimic the complexity of real-world scenes. Research into AI for RTS games are thus interesting game AI researchers, as developing advanced game-playing agents for enhancing playing experience in these environments is a hard problem requiring an integration of multiple disciplines. For instance, path-finding algorithms need to be incorporated in pre-battle scouting as well as in maneuvering units. Knowledge representation techniques might be used to model opponents' behavior and machine learning techniques for predicting strategies in order to generate counter strategies accordingly. Planning algorithms may then be used to produce efficient action plans to achieve different tactical goals. One key challenge in RTS game research is the incomplete information gameplay environment, whereby a player, whether human-controlled or computer-controlled, has only partial access to games states (e.g. location and terrain information) at any point of time.

Collecting and analyzing terrain information is commonly perceived to be a key foundation in forming RTS gameplay strategies, due to the vital information it contains. Terrain analyses have been used in strategy making for AI bots [17] (e.g. ambushing in narrow paths) [6] and in advanced path-finding algorithms (e.g. navigating units groups) [18]. Traditional terrain analysis research [6,16,17] commonly employ "cheating AI" by having full access to game map information. To our best knowledge, even though professional players are familiar with the popular maps, most novice players are unfamiliar with the game maps. Employment of "cheating AI" has been known to affect player experiences in a negative way [12].

We hence propose to develop an AI algorithm that obeys the same discovery rules as human players. This accounts for the fog-of-war mechanic present in most RTS games, which human players are subject to. As the preliminary exploration of this problem, the focus is on scouting spatial environment, where there is no enemy. In this article, we present a hierarchical multi-criterion decision-making (MCDM) scouting algorithm adapted from robotic exploration research [4], which is able to collect map information and recognize features in a plausibly fast manner by optimizing travel distance. In particular, the next-best-view (NBV) scouting strategy framework [2] is employed to structure our scouting algorithm. Within the framework, a hybrid map representation method, as well as a hierarchical candidate-position-identification mechanism is developed. Correspondingly, we also developed a multi-criterion-based candidate-view evaluation strategy. Our scouting algorithm contributes to creating comprehensive scouting agents, which are able to handle complex scouting tasks.

In the rest of this paper, we investigate related work on scouting strategies in both RTS game research and robotic research in section 2. Section 3 details our map representation method and exploration strategy. Section 4 describes our algorithm implementation and the experimental process. Finally, the merits and limitations of our algorithm are analyzed based on the results, and future research directions are outlined.

2. RELATED WORK

Building map information incrementally by gathering spatial environment data is a new approach for analyzing game terrain.

The challenge is how to navigate scout units in a right way, in other words, gathering more terrain data, in a short time, and avoiding damage. Potential field technique [9] has been used to deal with fog of war in Wargus[1] (a clone of WarCraft II[2]), to reveal the covered map. [15] presents a heuristic navigation tactic for scout units in collecting opponents' information. They devised a navigation method where a scout walks around the enemy base. There is still no accepted algorithm to solve unknown territory detection problem in RTS game fields.

The primary goal of scouting strategies is to collect the spatial data in a certain area. It is similar to robotic exploration of unknown terrain. For robotic research, the problem that computing an exploration path is a sub-field of the area-mapping problem, in which a robot equipped with a detection sensor with limited visible range, explores in an unknown planar environment to completely collect all the map information [5]. Since searching an optimal path for a map-coverage robot (i.e. start from an initial point, then completely explore the map space and go back to the origin point) is still a NP-problem, a number of algorithms are developed to approximately fulfill the task [3]. For instance, wall following strategies are simple ways to collect segments of the movement space border, which are presented in [13]. [11] presents a trajectory-based exploration strategy by constructing Voronoi diagrams. This solution relies on the pervasively distributed obstacles in the exploration space. Due to the incomplete knowledge of the space, there are many uncertainties in planning a path within several steps. A promising approach is to select the NBV in each step, where less distance is cost, while a big step is pushed forward to achieve the final goal (i.e. collecting more territory information). Normally, the NBV is chosen from either observable positions in current view or explored locations in previous steps. A coverage-map-based strategy is presented in [20], which formulates the map into occupied grids with a probability model. [2] presents the theoretical aspects of the criterion for determining the best observation positions, in which entropy theory is employed to calculate the expected information gathering. A MCDM strategy for choosing NBV is presented in [4] by using Choquet Integral [7] to combine criterion utilities. [21] presents a one-step-look-ahead strategy by generate a search tree from candidate positions during exploration. [14] formulates finding exploration paths in planar grid environment as a search problem, in which the occupation state of global grids is testing when doing next step planning. As investigated, frontier-based map representation method is proved to be an effective way to filter candidate positions for evaluation [1,2,4,8,10,21]. Comparing to grid-based map presentation, choosing potential next points along frontiers intuitively provides more chances to gather knowledge of unknown areas. There is, however, a big space to improve the map representation-based candidate-selection strategy. Due to the inherit constraints of video games, current exploration strategies still cannot meet requirements of scouting tasks for RTS games. Being forced to do scouting in the complex RTS game space makes it more challenging.

3. METHOD

We present a hierarchical MCDM-based scouting strategy for executing map-scouting tasks below. The scouting agent acts as the decision maker and procedure information collector by interacting with the simulation system. A hierarchical frontier-based mechanism is developed before making decisions to improve the efficiency of the algorithm.

3.1 Environment Representation

The exploration strategy works on two-dimensional environments. In this article, we employ a multiple map-representation methodology (i.e. the combination of Grid-based, segment-based and feature-based) to re-organize map-information for exploration agents to understand the environment, as well as provide search clues.

3.1.1 Grid-based Representation

The scouting unit is set as a ground unit in our simulation system. The assumption of it inherits from the main stream of commercial RTS games, i.e. it is unable to ignore obstacles when moving. A cone-shaped spatial patch surrounding the unit is revealed when it moves into an unknown area. The localization and map detection are concurrently conducted. Each detection activity in an unexplored region generates a map patch. It is, then, merged into a global map for incrementally build a complete representation of the environment.

In order to accurately note the unknown areas and the explored part, the grid-based map representation methodology is chosen. An A* algorithm is used for navigation, when the destination is decided in each step. The grid-based method divides game maps into square tiles using high-resolution grids. Each cell is marked as: unknown, free-movement or occupied. Unknown tiles refer to the cells that have not been explored. Players do not know whether they are walk-able or occupied by solid obstacles or other game units. Free-movement means that cells are able to be walked through. Otherwise, if a tile is flagged occupied, the tile is un-walk-able. The recon unit only walks through walk-able grids. An unknown cell is changed into walk-able or un-walk-able cell, when it is observed by a scout, i.e. it spatially falls into the unit's visible range. Game areas are distinguished and thereby are noted for both scouting and path-finding purposes.

3.1.2 Segment-based Representation

The frontier-based scouting algorithm relies on recognition of the areas' boundary. They are free-movement areas, unknown areas and occupied areas. Segment-based representation, then, acts as the boundary identifier. Areas are presented by boundary polygons composed of line segments. Globally, the frontiers, which separate unknown areas and unexplored areas, are identified by an algorithm that recognizes polygons from the border of detected regions [16]. The algorithm converts the two-dimensional exploration array into a geometric polygon-boundary representation. This process is also called vectorization. In each scouting step, the vectorization process is conducted when the scouting units arrive at the previous NBV. Hence, the new frontier vertex set is updated before evaluate the next NBV. Normally, some frontier segments are overlap with obstacle segments. As a requirement of candidate evaluation, the overlap parts need to be identified. In other words, frontier segments, which are on the boarder of obstacles, are recognized before evaluation. An analysis approach on its 8-trajectory cells is, thus used, to deduce whether a frontier vortex is on obstacles or on unknown parts.

[1] http://wargus.sourceforge.net/index.shtml

[2] http://us.blizzard.com/en-us/games/legacy/

3.1.3 Feature-based Representation

Compared to the general real or virtual-life scenarios, most RTS game maps normally have special elements. These elements play important roles in game playing. The development of game situation, basically, depends on the reconnaissance of these featured elements. For instance, gases and minerals provide economical support to StarCraft game-play. Given that, all the bases are built in mineral areas in StarCraft. Similarly, neutral campsites, stores and taverns, in WarCraft III, affect game-play by recruiting neutral heroes or offering special properties. Thus, it is necessary to gather information of these elements when conducting exploration tasks. We define these elements as special objects, which are featured with different geometric and functional properties. We assume that an object is completely detected when over 90% of its area is explored. Then, all of its property data is delivered to the scouting agent. The case follows the most situations that human players act in, who have the basic domain knowledge of the RTS game. When most parts of a special object are revealed to a human player, he is able to reasonably predict what the object is and what properties it has. These objects are presented as circular slices. The explored patches of special objects are maintained by the scouting agent during scouting.

3.2 Algorithm Framework

The general process is different from traditional path-planning algorithm. The scouting algorithm is not aiming to tackle the issue that finding a path from the initial location to the goal, but to manage the overall reconnaissance task.

Our algorithm follows the framework described below [2]:

a) A scouting unit perceives the surrounding environment.

b) The map patches perceived are integrated into the map representing systems.

c) Potential next positions are identified from map representation systems, according to a specific identification strategy.

d) The potential positions are evaluated with multi-criterions.

e) An optimal position is chosen as the goal of next movement after the evaluation.

f) The unit goes there and starts from a).

The hierarchical frontier-based potential position identification strategy is to address the issue in step c), while the MCDM-based evaluation approach is used to handle step d).

3.3 Candidate Position Identification

From the general exploration framework described above, we can abstract that the essential methodology of exploration is to find an optimal position for next movement. The definition of next position is a two-layer process. The first one is recognizing candidate positions from the game map. The last is to select an optimal position from the candidate positions accordingly. This section supposes to handle the former issue.

There are two reasons motivating us to pursuit the direction of optimizing the candidate position identification strategy. One aims to promote the computation efficiency of decision making. In the general NBV strategy framework, it is necessary to have a step-look-ahead for evaluating the information expected to gain in the possible next position. Computational resources (e.g. time and CPU) need to be assigned to the evaluation activities, whichever specific algorithm is used. Obviously, for utility-based evaluation approaches, the cost of computational resources increases, when the number of utilities goes up. The circumstance becomes severe, if candidate positions are massive in large-scale game maps. Another reason is to decrease the travel distance, moving from current scout's location to the NBV. It is not avoidable to travel a long distance for scouting an unknown region, even if areas around the scout unit have not been completely explored, when all of the global frontier vertices are taken into NBV evaluation in each step. This leads to the result that areas, which were temporarily given up to be detected, should be revisited in the later scouting process with travelling more distance. Even though, some NBV strategy [2,14,20] consider the travel cost during evaluation, a candidate position with rich expected information in a relative long distance away is commonly chosen. Alternatively, if the impact factor of distance cost is increased manually, the possibility of unreasonable prediction for the information gained in next potential positions goes up.

After considering the above, a hierarchical candidate position identification strategy is proposed. Motivated by the idea of level of detail (LOD), the scouting space is divided into several levels from the area closely surrounding the scouting unit, which is defined as level one, to the entire map.

In this article, we define four levels, namely, three local levels and a global level (see figure 1). Area levels are used to filter vertices on frontiers. For the filtering process, all the frontier vertices are traversed in the first step. Then, they are categorized into two different vertex sets. Vertices with an odd index number are pushed into an odd vertex set. Similarly, vertices with even index number are put into an even vertex set. Then, elements in the odd vertex set are processed. To be specific, points, which are located in the range of level three, are added into level-3 point set. Following the same rule, vertices are pushed into level-1 point set and level-2 point set separately.

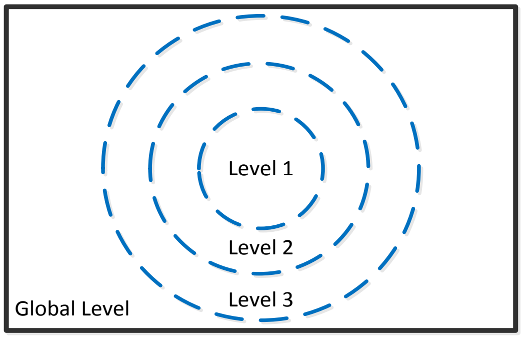


Figure 1. Hierarchical Filtering Levels.

After arranging frontier vertices into different level sets, candidate positions are selected accordingly. Each selection begins from the level-1 point set. In each step, the NBV is expected to be generated from the first level. If there are no points in level-1 set, the level-2 point set is processed for calculation. The rest can be done in the same manner.

The hierarchical model contributes to improve scouting performance as well as computational efficiency. Since local candidate viewpoints have higher evaluation priority, they are more likely to be completely explored. It is effectively avoiding the phenomenon of blindly travelling among unknown regions. Furthermore, the pre-processing of frontier vertices leaves less, but of higher quality (i.e. less travelling cost and more information gained) positions in each evaluation computation.

3.4 Candidate Position Evaluation

When considering the computation of potential-position evaluation, we borrow ideas from heuristic search algorithms. To gather more information costing less time or resources (travelling distance), the algorithm needs to balance between these two aspects. Hence, two components are developed, which are heuristic component and utility component separately.

3.4.1 Heuristic Component

A distance-based heuristic function is chosen. A Euclidian distance-based computational approach is employed. This real-distance-estimation method utilizes the same approach as we used in the A* path-finding algorithm for decreasing the computational complexity of candidate position evaluation. In order to weight the heuristic and utility components, the following equation is developed:

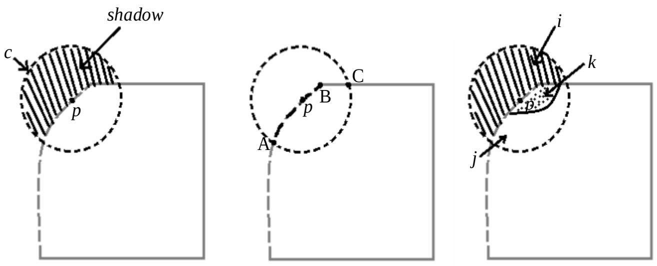
$$h(p) = e^{-[d(c,p)-r]} = e^{r-d(c,p)} \qquad (1)$$

where *d(c, p)* represents the estimating distance from current position *c* to the candidate position *p*. *r* denotes the radius of the scouting unit. This equation constraints the heuristic value to be between 0 and 1, and also guarantees that nearer candidate points acquire higher heuristic value, because the variation of the distance and the exponential function value follows an inverse proportion trend. The equation (7), discussed in next section, introduces *α* and *β*, which are used as weights for heuristic component and utility component respectively.

3.4.2 Utility Component

Effective data in RTS game circumstances can be categorized into three types. They are walkability of map tiles, outline of obstacles and special game elements. Technically, the knowledge of walkability of map tiles helps the game system render path-finding computation unlimited. It includes not only the troop maneuvering in later, combat scenarios, but also further scouting. The outline of obstacles plays a pivotal role when helping game-playing bots to functionally divide game maps into different regions. To our best knowledge, the up-to-date methodologies normally separate map space into free-movement regions (allowing large groups of troops move through side-by-side, or building extension locations on), narrow corridors (where ambushing always happens), and corridors. These region divisions are foundations for making strategies as well as predicting opponents' possible strategies by just considering spatial factors. Special game elements vary in different RTS games. As discussed above, they allow players developing special tricks, which in turn influence situations.

Then, the evaluation of information gained for these three types of data is introduced. To unify the value of each utility, the circumstance of each criterion is presented by percentage. *i*Grid(*p*) presents the possible un-walk-able cells gathered in the candidate position *p*. As shown is figure 2 (a), point *p* is the candidate position, while circle c illustrates the edge of scout's visible range if it is located on *p*. Cells in the shadow area are expected to be gained. The equation to calculate *u*(*i*Grid) is:



(a) *i*Area(*p*) (b) *i*Seg(*p*) (c) *i*Fea(*p*)

Figure 2. Information gain estimates with different criterions.

$$\mathrm{u}(iGird) = \frac{areaof\,(shadow)}{areaof\,(c)} \qquad (2)$$

In the second estimate, the amount of potential edge line that is visible in position *p*, is computed by *i*Seg(*p*). We assume that the frontier line segments, which are falling in the visible range of scout units, are obstacle segments. Figure 2 (b) illustrates that AB is a frontier line segment, and BC is a line segment of obstacles. The computation equation is:

$$\mathrm{u}(iSeg) = \frac{lengthof\,(AB)}{lengthof\,(AB) + lengthof\,(BC)} \qquad (3)$$

For the third elements, the expectation of obtainable game features in position *p* is predicted based on area of gathered features, which fall within the visible range of *p*. It is illustrated by *i*Fea (see Figure 2 (c)). The computing follows the equation:

$$\mathrm{u}(iFea) = \frac{areaof\,(k)}{areaof\,(j)} * \frac{areaof\,(i)}{areaof\,(i) + areaof\,(j)} \qquad (4)$$

where area *j* includes area *k*, and *k* is the patch of explored features in the current view point, while *j* is the explored area. i is the area, which is expected to be revealed in position *p*. A weight-based utility combination approach is used, which is demonstrated by:

$$u(p) = \sum_{i=1}^{n} u_i(p) * A_i(p) \qquad (5)$$

where $u_i(p)$ means the utility value of candidate position p with criterion *i*, $\mathrm{A}_i(p)$ is the weight of each utility. It satisfies the formula below:

$$\sum_{i=1}^{n} A_i(p) = 1 \qquad (6)$$

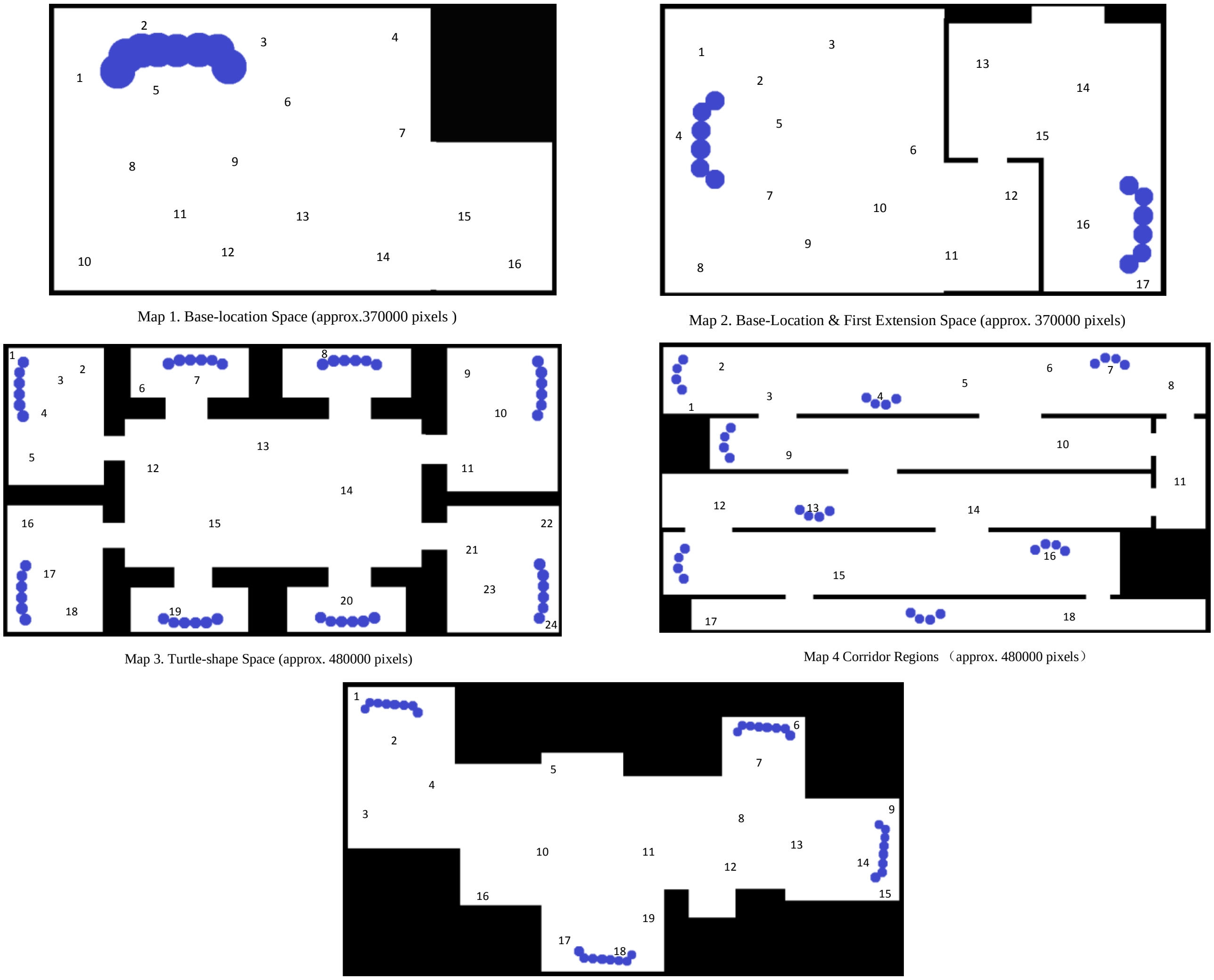
3.4.3 Combination of Components

A linear model is developed to combine the two components. It is illustrated by:

$$f(p) = \alpha * h(p) + \beta * u(p) \qquad (7)$$

The parameters (*α* and *β*) mean the weights that the two components have separately in the summary evaluation value. Their values satisfy the equation:

$$\alpha + \beta = 1 \qquad (8)$$



Map 1. Base-location Space (approx.370000 pixels)

Map 2. Base-Location & First Extension Space (approx. 370000 pixels)

Map 3. Turtle-shape Space (approx. 480000 pixels)

Map 4 Corridor Regions （approx. 480000 pixels）

Map 5. Extensions Regions （approx. 480000 pixels）

Figure 3. Game maps used in experiments (numbers represent starting locations).

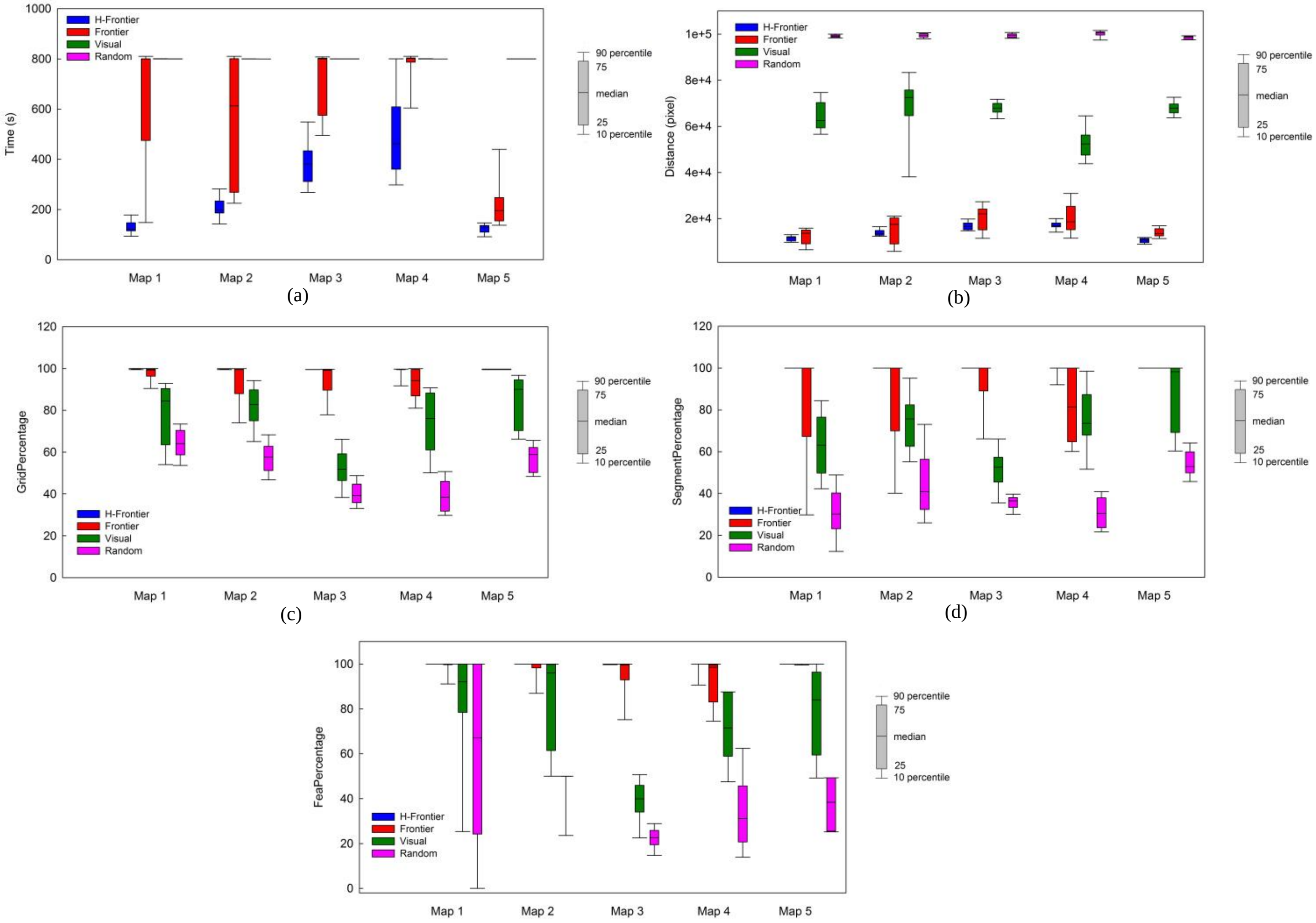
4. Experiment

To test our scouting strategy, we developed a simulator in C++ using *openframeworks*[3]. We collected 45 game maps from three commercial RTS games (StarCraft Broodwar, StarCraft II and WarCraft III), analyzed them, and extracted five common patterns: base-location pattern, base-location and the first extension pattern, turtle-shape, corridor and extensions.

Base-location pattern is fundamental to RTS games. This pattern represents the starting area, where the player’s base is located. To make it easier to be defended, it is often surrounded by obstacles or un-walkable terrain elements, such as seas and cliffs. A passageway, often referred to as a choke area, connects this region to the rest of the map. The first extension pattern is a resource-rich area that is connected with the base-location through a short corridor. It is often explored in the first several minutes of gameplay. The turtle-shape pattern defines terrain where there is a large free-movement space in the centre of the region and small regions containing resources connected to the main region through narrow corridors. A corridor pattern is a corridor-shaped common area, where narrow passageways are twisted. The extensions pattern is used to summarize a specific type of terrain frequently observed in WarCraft III. A group of semi-open extensions (i.e. regions rich in resources close to the base-location) are connected by an open space. This inner-connected large area is normally reachable from other regions through one or two narrow passageways. We have created abstract maps (see figure 3) based on these patterns. In these abstract maps, black areas represent walls and obstacles, while white areas represent free-movement spaces. Resources are represented by blue polygons. The purpose of the experiment is to test the ability of our strategy in scouting spatial environment. Hence, no enemy is settled in these maps.

For all experiments we chose the sight range of scouting units as 40 pixels, which forces recon units to make a significant number of steps to complete the exploration. Scouting simulation is terminated when the map is 99.5% explored or when the allocated time (800s) for exploration runs out. This percentage is enough to reflect the completeness of space information gathering.

[3] http://www.openframeworks.cc/



(a)

(b)

(c)

(d)

(e)

Figure 4. Performance of strategies in different criterions.

It is meaningless to compare the remaining 0.5% exploration in evaluating scouting algorithms. We describe the four strategies used in our experiments below.

A. Random Strategy

The candidate positions are randomly selected from explored areas, and the next movement position is randomly selected from the candidate positions i.e. the evaluation function is not used. This strategy represents an uninformed agent intended as a worst-case strategy for exploration.

B. Visual Strategy

The visual strategy is in the family of NBV strategies. Candidate positions are chosen from current scout's visible range, and often along the edge of visual range to maximise new area. Candidate positions are evaluated using the same MCDM strategy as the Hierarchical-Frontier Strategy described below. Integrating this strategy gives us a chance to compare the performance of scouting algorithms between frontier-based and non-frontier-based.

C. Frontier Strategy

This strategy is a modified version of the González-Baños and Latombe's exploration strategy (GB-L strategy) [8] and used to determine the performance contribution that can be attributed to the hierarchical position filtering mechanism. As part of MDCM, candidate positions from frontier vertices are identified and evaluated by the following formula:

$$f(p) = A(p) * \exp(- \lambda * L(p))$$

where A(p) is an estimate of the unexplored area visible from p, L(p) represents the real distance from the current location of scouting unit to the candidate position p. λ denotes the weighting between exploring large areas and travelling less distances. The value of λ is set to 1/300, meanwhile the value of A(p) is constrained to [0~1], and the value of L(p) is in [0~1000]. That helps to balance A(p) and L(p) in a same magnitude as well as to keep their contribution for the total evaluation value in a proportion of 2/3. The value of λ is empirically chosen and varying this value is part of our future research. In terms of unexplored area estimation, two criterions are taken into account. They are unknown map-grid gathering and obstacle-segment collecting. This strategy is a representative example of the state-of-art in NBV-based strategies in solving robotic exploration problems.

D. Hierarchical-Frontier Strategy

This strategy represents our novel contribution. It employs Hierarchical-Frontier (H-Frontier) candidate identification strategy to filter potential positions before evaluation. In terms of the weights of position evaluation, we emphasis more on grid gathering and segment gathering, since these two kinds of information are used to re-construct the outline of the map territory. Then, the parameters of utility component (Equation (5)) are set as: 0.4, 0.4 and 0.2 for A(iGrid), A(iSeg) and A(iFea) respectively. Compared to distance travelling, we give priority to information gathering in the next view. In equation (7), the weights we have chosen give priority to information gathering over travelling less distances with α set at 0.4 and β set at 0.6.

For each map, we have chosen 16-24 possible starting positions (shown in Figure 3), determined by the complexity of the map. Each game map is evaluated using the four different strategies with all the different origin positions. Box plots, shown in figure 4, aggregate the results from simulation runs with different origin positions. Five criterions used to evaluate the performance of the strategies are: time cost (seconds), travelling distance (pixels), percentage of free-movement grids collection, percentage of obstacle segments, and percentage of game features (such as resources) gathered during scouting. The horizontal line in the box plot indicates the median value, box-boundaries are the 25th and 75th percentiles, and whiskers are the 10th and the 90th percentiles. The percentage plots illustrate that the H-Frontier strategy is the only one that is able to statistically complete scouting tasks in all the game maps and for all the origin positions. Figure 4(c), 4(d) and 4(e) illustrate that the 10-percentile value of map information-gathering percentage for H-Frontier strategy is over 99.5% in map 1, 2, 3, and 5, and that the 25-percentile value is over 99.5% in map 4. While the 10-percentile value of Frontier strategy is just over 99.5% only in map 4. The other two strategies perform even worse – rarely complete scouting tasks. In terms of time spending, the H-Frontier strategy performs significantly better than other strategies in each map. Almost all the time spending for H-Frontier is less than other three strategies in all the maps (Figure 4(a)). Furthermore, the H-Frontier also travels less distance in completing scouting tasks in all of the maps (Figure 4(b)).

For few cases in maps 1, 2, 3 and 4 the Frontier strategy travels less distance than the H-Frontier Strategy, but in all of these cases the Frontier strategy fails to complete the scouting mission. As demonstrated by the percentage plots, the 90-percentile value of distance of H-Frontier Strategy is even lower than the median value of Frontier Strategy in maps 1, 2 and 3. To summarize, the Hierarchical Frontier Based Strategy we presented performs better than others for scouting in RTS game environments based on the five different criterions we have discussed.

5. RESULTS AND DISCUSSION

In the experiment, we compared candidate position identification strategies, and demonstrated that our algorithm performs better overall. To be specific, frontier-based strategies achieve overwhelmingly better results than the other two strategies. Among them, due to the hierarchical position filtering mechanism, the H-Frontier performs better than Frontier strategy in every aspect, which means the former, is able to collect more spatial information whilst travelling less distance during less time. In most position evaluation processes, the H-Frontier only computes the utility value of local positions, while the Frontier strategy has to calculate it globally. This results in saving time on evaluation computation. Additionally, since the local candidates are given high priority, the phenomenon of going back-and-forth in long distance is eliminated. It is common that a candidate position that is far away from current spot with potential in gathering more environment data is selected as the NBV. The fact is, however, that the scout has to travel back again for the completeness of exploration. In this experiment, the scout settings (such as α, β and the sight range for the scout) have been kept constant. We are currently running additional experiments varying these settings in order to gather statistical results and test the stability of our solution. Another area we are exploring is how position evaluation strategies can be compared when combining different estimate factors.

6. CONCLUSION

We presented a Hierarchical Frontier MCDM-based scouting strategy to perform scouting tasks in RTS games by sensing its environment as part of a terrain analysis agent. A novel hierarchical-position-identification strategy considering factors that affect RTS game playing was developed, implemented and tested against similar strategies. The results show that our strategy performs better than others in the complex RTS game environment. Our results should also benefit research in robotic exploration strategies, although we have not tested it for that domain. As part of our future work, we will be examining the effect of enemy units and exploring the terrain using multiple units as part of an RTS game.

7. ACKNOWLEDGMENTS

The authors would like to thank the financial support provided by the China Scholarship Council, the University of Technology, Sydney, and the Centre of Human-Centred Technology Design.

8. REFERENCES

[1] Amigoni, F. 2008. Experimental evaluation of some exploration strategies for mobile robots. *IEEE International Conference on Robotics and Automation*. 2818-2823.

[2] Amigoni, F., & Caglioti, V. 2010. An information-based exploration strategy for environment mapping with mobile robots. *Robotics and Autonomous Systems*, 58,5, 684-699.

[3] Arkin, E. M., Fekete, S. P., & Mitchell, J. S. 2000. Approximation algorithms for lawn mowing and milling. *Computational Geometry*, *17*,1, 25-50.

[4] Basilico, N., & Amigoni, F. 2009. Exploration Strategies based on Multi-Criteria Decision Making for an Autonomous Mobile Robot. In *ECMR*. 259-264.

[5] Choset, H. 2001. Coverage for robotics–A survey of recent results. *Annals of mathematics and artificial intelligence*, 31,1-4, 113-126.

[6] Forbus, K. D., Mahoney, J. V. & Dill, K. 2002. How qualitative spatial reasoning can improve strategy game AIs. *Intelligent Systems*, 17, 4, 25-30.

[7] Grabisch, M., & Labreuche, C. 2008. A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. *4OR*, *6*,1, 1-44.

[8] González-Baños, H. H., & Latombe, J. C. 2002. Navigation strategies for exploring indoor environments. *The International Journal of Robotics Research*, 21,10-11, 829-848.

[9] Hagelback, J. & Johansson, S. J. 2008. Dealing with fog of war in a real time strategy game environment. CIG '08. IEEE, 55-62.

[10] Holz, D., Basilico, N., Amigoni, F., & Behnke, S. 2010. Evaluating the efficiency of frontier-based exploration strategies. *Joint 41st International Symposium on Robotics and 6th German Conference on Robotics* 2010. 36-43.

[11] Kim, J., Zhang, F., & Egerstedt, M. 2010. A provably complete exploration strategy by constructing Voronoi diagrams. *Autonomous Robots*, 29,3-4, 367-380.

[12] Laird, J. & VanLent M. 2001. Human-level AI's killer application: Interactive computer games. *AI magazine* 22, 2, 15-25.

[13] Lee, D., & Recce, M. 1997. Quantitative evaluation of the exploration strategies of a mobile robot. *The International Journal of Robotics Research*, 16,4, 413-447.

[14] Li, A. Q., Amigoni, F., & Basilico, N. 2012. Searching for Optimal Off-Line Exploration Paths in Grid Environments for a Robot with Limited Visibility. In *AAAI*. 2060-2066.

[15] Park, H., Lee, K., Cho, H.-C. & Kim, K.-J. 2012. Prediction of early stage opponents strategy for StarCraft AI using scouting and machine learning. In *Proceedings of the Workshop at SIGGRAPH Asia*. ACM, 7-12.

[16] Perkins, L. 2010. Terrain analysis in real-time strategy games: an integrated approach to choke point detection and region decomposition. *AIIDE '10*. AAAI, 168-173.

[17] Pottinger, D. C. 2000. Terrain analysis in real-time strategy games. CGDC '00

[18] Preuss, M., Beume, N., Danielsiek, H., Hein, T., Naujoks, B., Piatkowski, N., ... & Wessing, S. 2010. Towards intelligent team composition and maneuvering in real-time strategy games. *Computational Intelligence and AI in Games, IEEE Transactions on*, 2, 2, 82-98.

[19] Rocha, R., Ferreira, F., & Dias, J. 2008. Multi-robot complete exploration using hill climbing and topological recovery. *IEEE/RSJ International Conference on Intelligent Robots and Systems.* 1884-1889.

[20] Stachniss, C., & Burgard, W. 2003, August. Exploring unknown environments with mobile robots using coverage maps. In *IJCAI*. 1127-1134.

[21] Tovar, B., Muñoz-Gómez, L., Murrieta-Cid, R., Alencastre-Miranda, M., Monroy, R., & Hutchinson, S. 2006. Planning exploration strategies for simultaneous localization and mapping. *Robotics and Autonomous Systems*, 54, 4, 314-331.