

Large-scale mapping and navigation in virtual worlds.

RSMG Report 5

Katrina Samperi
k.samperi@cs.bham.ac.uk

July 14, 2011

Supervisors: Russell Beale and Nick Hawes
RSMG Members: Richard Dearden and Bob Hendley

Abstract

The problem we are trying to solve is how to enable intelligent mobile agents to generate and use maps of large-scale, dynamic, unstructured environments. In contrast to previous research in agent mapping, we are investigating the difference made when looking at increasingly large, complex environments. This report covers progress in solving this problem. First we look at methods of map representation, and methods for evaluating any proposed changes. We evaluate the map through its use, and so we look at how the agent can use a given map to plan routes in different sized environments. We gather baseline information about the time taken to generate a probabilistic road map and plan routes using this, which can then be used when comparing map segmentation and other representations. We conclude that while it is possible to create and use probabilistic road maps, it still takes too long for the agent to create the initial map. We also discover the need to test the road map generation methods in different types of environment. Finally we look at the original research plan and discuss any changes to be made.

1 Introduction

The problem we are trying to solve is how to enable intelligent mobile agents to generate and use maps of large-scale, dynamic, unstructured environments. In contrast to previous research in agent mapping, we are investigating the difference made when looking at increasingly large, complex environments. Most recent work in agent mapping focuses on small-scale, indoor, office-like environments. When dynamism is considered it is at the expense of size, and large-scale environments considered are usually static. We ignore the need to implement localisation and accurate sensing by considering the problem for virtual agents existing in the large-scale, persistent environment of Second Life.

We can split the problem of how to enable intelligent mobile agents to generate and use maps of large-scale, dynamic, unstructured environments into several areas. First, how do we best represent a map of large scale, dynamic environments? This map must be generated quickly and accurately as the agent will be required to do this online, rather than requiring offline preprocessing. This means speed is an important factor for any representation, generation and maintenance solution. Once the agent has a map the second question is, how does the agent update and maintain this successfully? Our current work has focused on the first of these questions, namely how to represent the map.

In the remainder of this report we shall discuss recent work relating to map representation and how to define a good map. Following this we evaluate the probabilistic road maps by comparing their generation in increasingly large virtual environments and how different techniques for doing this compare. We look at the generation time for the map, route

planning time and length of the shortest routes to establish a base line for the agent. By establishing this base line we can compare any future changes to the map, such as segmentation, and state whether they have a positive or negative effect on the overall representation. The original research plan will be discussed and any changes added, along with a timetable for completion. Finally we give an overview of some of the new literature, relevant to this current work.

2 Recent Work

In order to test and evaluate the map representation we first needed to create a framework which would allow the agent to access the virtual world, collect information and use this to navigate. This has now been completed and has been used to generate object maps of 'real' virtual environments and navigate from one place to another while still online. More recently, this has been adapted to work with the CAST framework, and testing done to ensure it is stable enough to run long term experiments.

The agent system currently consists of:

- Object Map
- Probabilistic Road map Generation
- A* Route planner
- Trail recording
- Control component and file save / loading options

Once the framework was complete we started to look at the map representation and how to evaluate it. The map representation is a vital component of the solution to our problem. A good representation will allow the agent to generate a map quickly and accurately. The representation should be compact, requiring the least amount of space so that it can be stored efficiently in both internal memory and on disk. As the environment is dynamic, the map will need to be capable of being updated and maintained. The map should reduce the time needed for the agent to plan and follow a route and not require any lengthy offline or pre-processing before use. The type of map representation used will affect all of these issues and so it is important to look at each map in terms of how it is generated, how it will be used and how it can be maintained.

There are two options for the map when looking at large-scale environments. The simple option is for the agent to have a single object map encompassing everything the agent has ever seen. Using a single map will cause the storage space and processing time requirements to grow too large. The agent will face significant difficulty in using and maintaining the map in the long term. The alternative to this approach is to split the world into smaller segments, allowing the agent to consider each of these individually. As Second Life is split into regions of 256m³, a simple method of dividing up the world would be to use these boundaries. However, to allow the map to be used in all environments, it is more useful to split the map according to its features, rather than the technical constraints of the virtual world used.

Dynamism will, in theory, influence the best way to segment the environment. An ideal representation would not need to be regenerated every time there is a change. To enable this the agent will need to intelligently split the space into smaller segments to keep any regeneration as small as possible. How the segments relate to one another will be stored at the topological level, with the optimal map having the least amount of dynamism in the highest levels of the representation. For example, regions will change their position, relative to one another, on a much less frequent basis than a group of moving objects present in a smaller sub-region.

We can compare the different methods of segmenting space by looking at how each affects the generation, use and maintenance of the map. A good representation will confine changes to a single segment at a time. It will still allow the agent to plan routes crossing the boundaries between segments and not be too detrimental to the generation time of

the map. The trade-off between the added overhead of using several small maps and the benefits gained for planning and maintenance will need to be investigated.

To this end we have been looking at probabilistic road map generation. We have been gathering some base line statistics for an agent using a single map of the environment. These can then be used in the future to evaluate segmentation methods and alternative map representations.

3 Probabilistic Road maps

3.1 Motivation and Aim

Probabilistic road maps (PRMs) are a method of generating a topological map of an environment (Kavraki et al., 1998). They have been used in many different domains to plan a set of actions that must be taken to reach a goal. LaValle (2006) goes into more detail about the generation of PRMs and how each change can affect the overall map generated. We wish to use a PRM to allow the agent to plan a route from one point to another in a given environment. PRMs are considered easy to generate and useful when agents are attempting to navigate large scale environments. The success of planning and following these routes would allow us to evaluate the how good the current map representation is compared to a basic map. To do this we need to generate a set of baseline information about the type of PRMs that are generated in the specific virtual environments we are interested in mapping.

It is important that the PRM can be generated fast and accurately as this should occur online, rather than require offline preparation. A good road map will have a high connectivity and allow the agent to navigate from any two points in free space. We measure how well each road map is generated by comparing the time taken to generate the map, the time taken to plan a route, the average length of the shortest path and the coverage given by each approach.

There are several different factors that can affect the generation of PRMs.

- Size of the environment
- Number of objects present
- Placement of these objects (clustered together or significant distances apart)
- Number of points
- Distance of arcs
- Method for generating the points
- Method for determining if a point is free

We wish to investigate how changing the above factors will affect the generation of PRMs in the virtual environments the agent will inhabit. While doing this we wish to generate a baseline set of results for how long the agent takes to generate a PRM for each size of environment, and then use this as a comparison for how any future changes to the overall map representation positively or negatively affect the agent.

3.2 Method

For this set of experiments the agent is offline and is given a set of objects with which to create an object map. We wanted to look at this offline to ensure the object map created is identical between map generations. We can then vary the size of the environment, by giving it a different set of objects to consider. We will look at four different sized

environments from 128x128m up to 256x512m. Table 1 shows the different sizes of the environment and the number of objects present in each one.

Table 1: The number of objects in each size environment

| Second Life area | Point | |
|---------------------|-------------------------|-------------------|
| | Size of the Environment | Number of Objects |
| Quarter of a region | 128x128m | 166 |
| Half of a region | 128x256m | 401 |
| One Region | 256x256m | 1850 |
| Two Regions | 256x512m | 3700 |

We first calculated where the free space was in each environment. Figure 1 shows the different maps used. The white in the image is free space the agent can move into; blue is space blocked by an object. We then selected 8 points, attempting to place them as close to the corners and half ways points as possible in each environment. The agent then attempts to generate a PRM, adding points to the map until it can plan a route between any of the selected points. We deliberately select points which are able to be reached in each environment environment. Routes are planned using A* (Russell and Norvig, 2003).

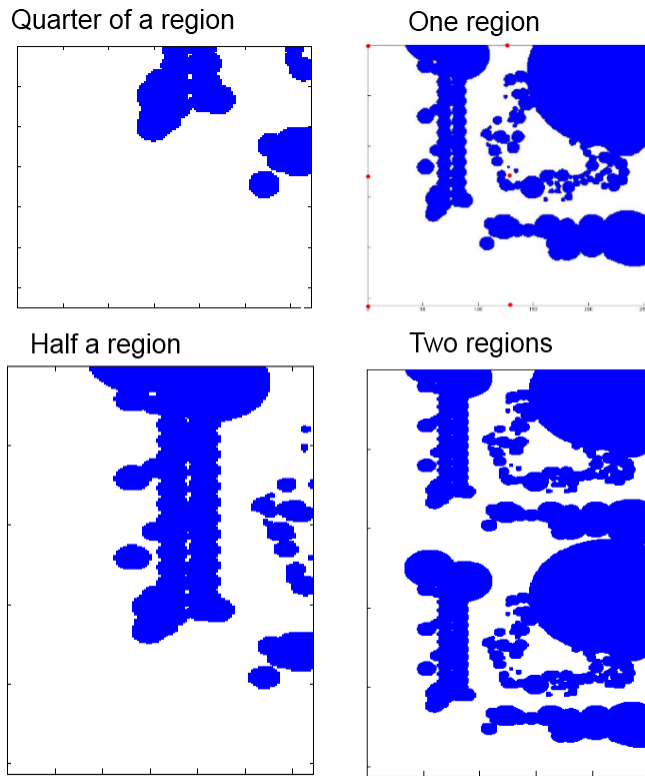


Figure 1: The environment used by the agent. White space is free and blue is blocked.

The point generation method used is one way of changing how the PRM is built. We want to look at three popular methods for generating points in free space. The first is a totally random approach. This is fast and requires no prior knowledge of the environment, but does not always give the best coverage. The second method is a grid based approach. Points are generated in a grid configuration with the distance between points reducing each round. By doing this, grid point generation gives a good coverage of the entire environment, but requires the checking of many more points than may be required. Also, without a very fine grid it can be impossible to create paths in non grid-based environments (a narrow, curving corridor for example). Finally we look at a quasi-randomised approach suggested by Branicky et al. (2001) called the Halton Sequence. Branicky et al. believed these had a positive effect on the coverage and generation of points, but Geraerts and Overmars (2002) claim it has little useful effect. For the first set of experiments we have kept the distance for neighbouring points being joined at 10m.

We evaluate the maps generated by looking at the overall time taken to generate a PRM that can be used to plan a route between all the selected points. A shorter generation time is preferable as the agent will be doing this online, rather than preparing the map in advance. We also want the PRM to reduce the time needed to plan a route and also which finds the shortest route between the set of points. We will look at whether a shorter generation time leads to a longer planning time, negating the benefits gained. Overall, we will get a set of baseline results for the best PRM that can be generated using a single map of the environment.

All the experiments are run on a single computer, under as close to the same circumstances as was possible. The machine was an Intel(R) Core(TM)2 Quad CPU (2.40GHz) running Scientific Linux version 2.6.32 with 2gb RAM.

3.3 Results

We first looked at how the size of the environment and point generation method used affect the overall generation of the PRM. The maximum distance two points can be considered neighbours in the random and halton approaches was 10m apart. In the grid approach the distance was reduced as the distance between points also reduced. In each environment the agent was given a set of eight points, roughly corresponding to the corners and midpoints to plan routes between. All the results gathered are shown in table 2. The numbers given are averaged over 20 runs.

Table 2: PRM generation time, averaged over 20 runs in different sized environments. The distance between points for Halton and Random generation was set at 10m and the line checking approach sequential. The numbers in brackets are the standard deviation for each average.

| Size | Point Gen | Iterations | | No. Points | | No. Arcs | | Avg Time in sec. | |
|-------------|-----------|------------|---------|------------|----------|----------|-----------|------------------|-----------|
| Quarter | Random | 8.95 | (3.47) | 352.4 | (144.67) | 1893.95 | (1835.35) | 13.02 | (13.16) |
| | Grid | 5.00 | (0.00) | 438 | (0.00) | 1755.00 | (0.00) | 10.19 | (0.24) |
| | Halton | 17.85 | (0.65) | 749.05 | (28.71) | 6521.90 | (543.31) | 61.24 | (5.67) |
| Half | Random | 22.95 | (5.90) | 845 | (217.57) | 5588.10 | (3153.98) | 319.88 | (186.96) |
| | Grid | 6.00 | (0.00) | 2818 | (0.00) | 58239.00 | (0.00) | 3478.71 | (17.56) |
| | Halton | 24.00 | (0.71) | 896.2 | (26.54) | 4928.05 | (313.90) | 290.79 | (18.28) |
| One region | Random | 42.00 | (10.70) | 1246.5 | (319.04) | 7571.15 | (4136.39) | 1739.81 | (979.24) |
| | Grid | 5.00 | (0.00) | 1176 | (0.00) | 4460.00 | (0.00) | 930.77 | (1.69) |
| | Halton | 30.85 | (7.93) | 919.35 | (244.30) | 2787.20 | (1354.31) | 811.67 | (484.10) |
| Two regions | Random | 83.30 | (20.57) | 2334.65 | (568.01) | 13997.15 | (6918.55) | 6766.20 | (3398.92) |
| | Grid | 5.00 | (0.00) | 2193 | (0.00) | 8231.00 | (0.00) | 3568.45 | (15.60) |
| | Halton | 82.05 | (0.02) | 2299.15 | (424.50) | 13214.30 | (5007.69) | 6411.28 | (2495.29) |

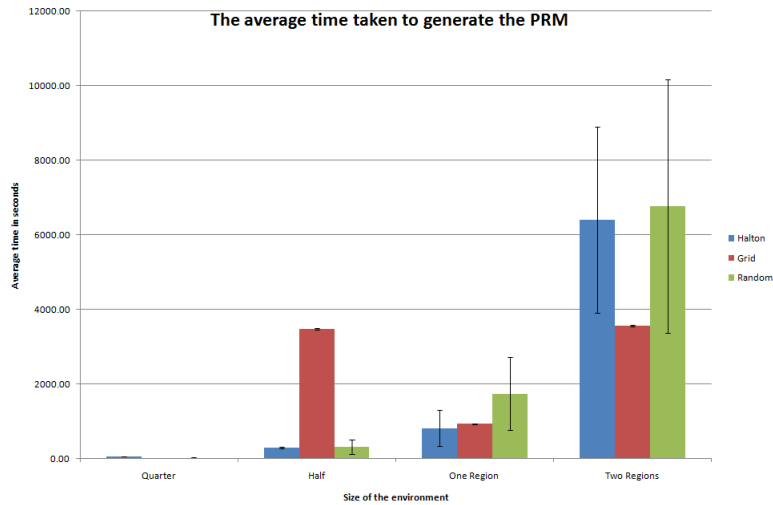


Figure 2: Average time needed to generate the PRM

The time required to generate the PRM can be seen in Figure 2. We can see from this that, unsurprisingly, as the size of the environment increases the time taken to generate a PRM also increases. Grid based is the most predictable result, but generates and tests far more points than the other methods. The halton approach generally takes less time to generate a point than the other two methods. An exception to this are in the very small, quarter sized environment where random was better, possibly due to the lack of blocking objects in this environment. Even when using the best point generation method, the time taken to create the PRM is still too long.

An interesting point on this graph is the grid point generation in a half sized region. The time required for this vastly outstrips the other two methods. This is due to the way grid points are selected and the location of the point to navigate to. The point requested for route planning was in a corner, blocked on three sides. To join this point to the PRM the grid approach had to go one iteration further than in the other maps, generating points only 2m apart. This increase in generated points can also be seen in Figure 4.

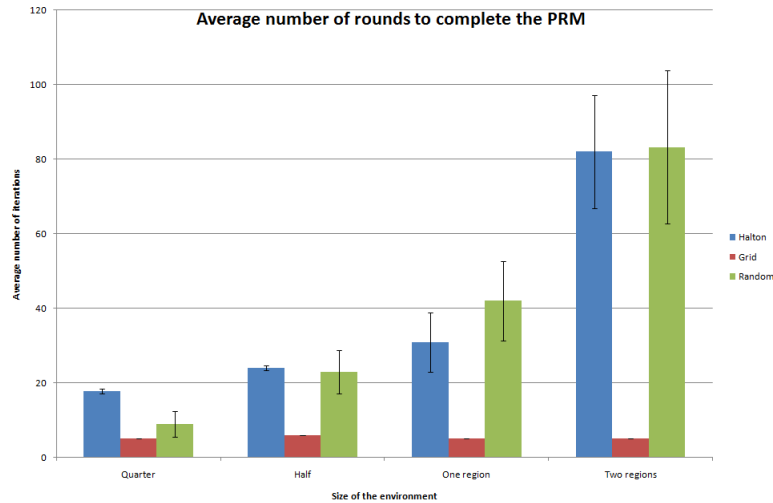


Figure 3: Average number of iterations required to generate a PRM that can be used to plan all the routes

The number of iterations required to complete the PRM is shown in figure 3. Grid generation has a static number of rounds required. This is due to the way grid generates the points. It is a deterministic approach, so the number and placement of the points is always the same between runs. However, grid point generation does have the disadvantage of creating a number of redundant points in each iteration. Halton and Random point generation require almost the same number of iterations, the standard deviation in these approaches is quite high. The environments we were looking at had a number of difficult points to be connected to the PRM. Many of the iterations in the random and halton approach will have been wasted as the map required a point to be selected in a very narrow area of free space to connect all the points.

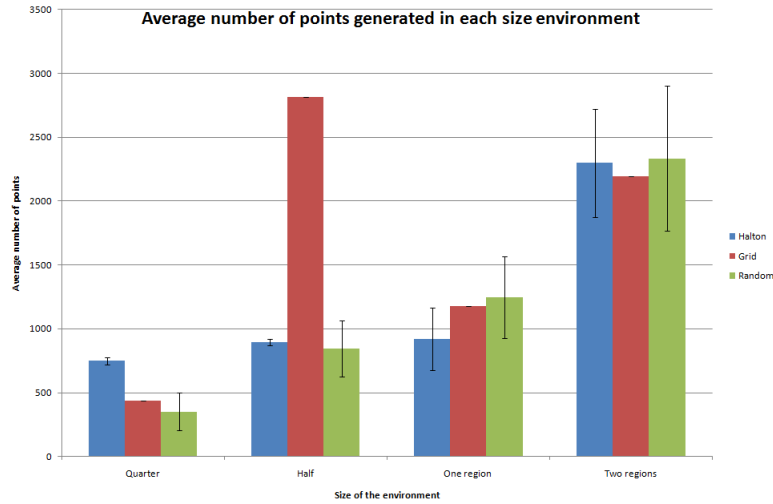


Figure 4: Average number of points generated against the size of the environment

The average number of points generated is shown in 4. For very small regions the random approach required the smallest number of points to generate the PRM. However, as the size of the environment increases, the number of points generated is higher using random point generation than the other two approaches. The quarter sized region has the greatest percentage of free space, and so as the environment becomes more constricted the random approach requires more time, iterations and points to be generated. An interesting point we can see here is that the number of points required to generate the grid point PRM in the half sized environment is much higher than any of the other PRMs. This environment also required six rounds to generate the full PRM, rather than five needed by every other grid point PRM.

3.3.1 Route planning

Table 3: The average planning time and length of the shortest route between points. 28 different routes were calculated in each map size

| Size | Point Gen | Shortest planning time | Shortest route length |
|--------------|-----------|------------------------|-----------------------|
| Quarter 4 | Random | 3.6% | 0% |
| | Grid | 92.8% | 67.9% |
| | Halton | 3.6% | 32.1% |
| Half | Random | 0% | 0% |
| | Grid | 0% | 89.3% |
| | Halton | 100% | 10.7% |
| One region | Random | 0% | 10.7% |
| | Grid | 3.6% | 27.6% |
| | Halton | 96.4% | 10.7% |
| Two regions | Random | 0% | 0% |
| | Grid | 100% | 100% |
| | Halton | 0% | 0% |

We were also interested in which approach generated a PRM that took the least amount of time to plan the routes and which of these were the shortest. The PRM should allow routes to be generated fast, and preferably, using the shortest path.

Table 3 shows the percentage time each point generation method achieved the shortest planning time, or shortest route length. From this we can see that in all cases the grid based point generation had the shortest route length. Half the time the grid point generation had the shortest planning time, the others being halton. The speed of planning is tied into the number of points generated, so where the number of points is less for halton, the planning time is shorter.

3.3.2 Coverage

Coverage is important in the point generation type. The PRM should be able to connect all points to the map, but without points being generated in all areas then this is not possible. We evaluated coverage by summing the number of points in an area $5m^2$ and using this to generate heat map images of the environment. Good coverage will have points in the correct places to link the destinations the agent has been given. The best solution will use the least number of points to do this. A bad solution will have a very high number of points in a small area that do not add anything to the overall map.

The resultant images for the single region environment are shown in figure 7. Grid is, unsurprisingly, very good for coverage. The entire map is sampled, in a regular way. This however generates many more points than may be required to plan the routes. Random point generation can have either good coverage, or very bad. The higher frequency of 'hot' colours in the heat map show there may be many more points generated than are required to plan the routes. The most even coverage for the three approaches is using the halton sequence. This has a good coverage of the map and does not end up with clusters of points in a small area.

We can also use the coverage to look at the anomalous result we got for grid point generation in the half sized environment. As could be seen in figure 2 grid point generation took far longer than any other method, and longer than in a larger sized world. This is due to the drawback inherent in grid point generation. One of the points selected was in the corner, next to two blocking objects. For this point to be connected to the graph it took a very fine grid (points two meters apart) to connect all the points. The coverage map, figure 6, shows the large number of points generated, with

large amounts of the environment orange, indicating around 60 points when all the coverage maps are added together. A large number of these points are redundant, as they were not needed to generate the other routes.

We have also included the images generated for the quarter sized environment (Figure 5) and the two region environment (Figure 8) for completeness.

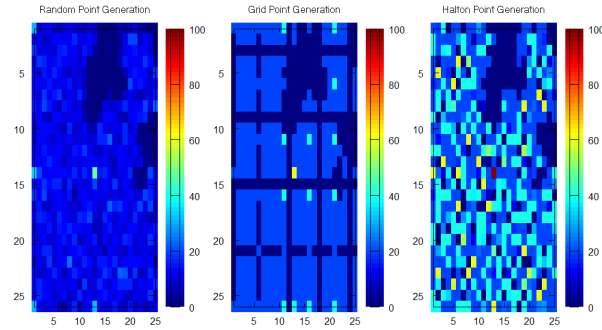


Figure 5: The point coverage for the quarter sized environment.

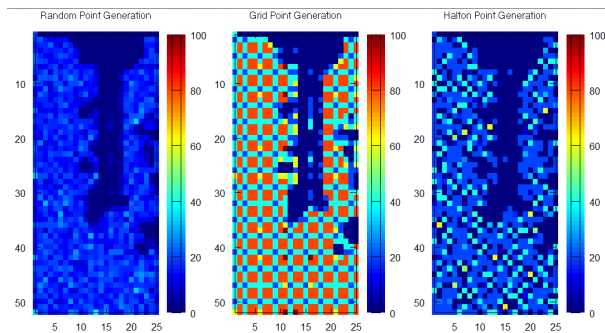


Figure 6: The point coverage for the half sized environment.

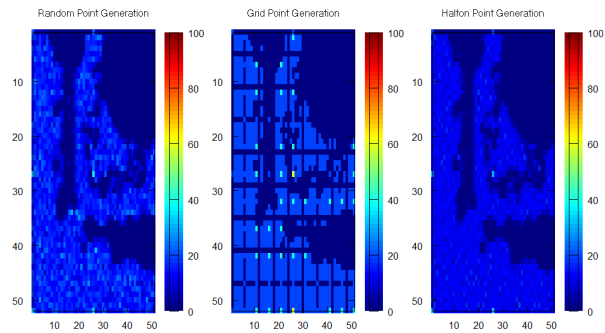


Figure 7: The point coverage for the one region environment.

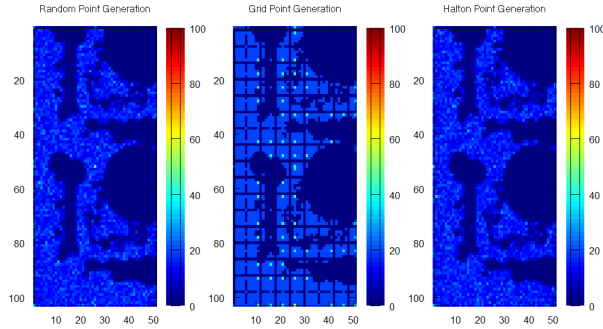


Figure 8: The point coverage for the two region environment.

3.4 Discussion

We defined a good probabilistic road map as being generated fast and accurately with high connectivity of points which allow an agent to plan routes between any two points in space. To do this the coverage must be correct. There must be enough points to connect the map, without creating too many redundant points. We found that coverage does not necessarily have to be evenly spread across the map, but generating a point in the correct place, such as in narrow corridors, is far more important.

We found in these preliminary experiments that all point generation methods are viable. In small scale environments the time taken to generate the PRM is very similar, but as the environment grows larger, grid point generation takes the least time. This is due to the very regular method of selecting the next point to add, rather than the pseudo, or quasi-randomised approach favoured by the random and halton approaches. Planning time is linked to the number of points in the PRM. The grid and halton generation methods created PRMs with the shortest planning time, but in all but a few cases, the grid approach created the shortest paths. Coverage was complete in all three approaches. The fastest to gain an even coverage of the whole map was the grid approach, but as can be seen, for hard to reach points many more points will be generated than may be necessary. Halton had, on average, a better coverage of the map than the random approach.

Even when using the best approach the time taken to generate the map is far too long for it to be successfully done with an online agent. We can use the numbers generated as a baseline to compare any future map alterations considered. We cannot draw any firm conclusions about the best type of PRM to use from these experiments. We saw that if the destination is in a confined space, the number of iterations and points required to generate the map creates many redundancies. We need to further test the road map generation methods in different environments to discover the best possible result.

We can however say that unlike Geraerts and Overmars (2002) states, the point generation method used does make a difference when generating maps of larger worlds. As the size of the environment increases we expect to see these differences become even more apparent.

There are still some further investigations we can undertake regarding PRMs in large virtual environments. The largest environment we looked at here was 256x512m, two Second Life regions. We had hoped to double this up and investigate how the generation time is affected, but there was not enough time to conclude these experiments. We had also hoped to vary the number of objects present in the environment, showing how the different number and placement affect the generation time. The very high generation time for grid point based PRMs in the half sized environment show that we need to consider many more types of environments before we are able to draw a set of conclusions. We can also look at setting the neighbouring distance between points as a factor of the environment size. This would mean that the size of the environment does not affect the connectivity of the points, only the obstacles.

The next step will be to look at segmentation of the environment. Clearly it is not feasible to use a single object map as it takes too long to generate the PRM and so plan a route. Segmentation should decrease the time taken to generate the PRM. We can look at whether this is accurate, or if the overhead of splitting the environment and moving from one segment to another is detrimental to the overall representation.

One of the proposed improvements for the overall map was to use trails as a source for generating PRM points and arcs. This has yet to be fully investigated, but preliminary testing show that while this does speed up point generation on a point by point basis, it also generates a large number of points clustered in a small area. The resulting map therefore takes a long time to build, while having a very limited coverage.

A potential compromise between the different point generation types would be to use sparse grid as the base of a PRM. This can then have the trail information added, once useful points have been extracted. The addition of trail points would give some routes through narrow and difficult to reach places. Finally, an 'on demand' PRM generation method could be included. As the agent is required to plan a route it starts adding points along that route attempting to connect the start to the destination. This 'on demand' method would be comparable to rapidly exploring randomised trees (Missiuro and Roy, 2006). By generating points to specifically fit the routes the agent should have a reduced generation time, though the map would need updating whenever a new route was required. We can also look at octree style, or adaptive grid methods for generating the grid points (LaValle, 2006).

Missiuro and Roy (2006) compares the use of Gaussian point generation and bridge obstacle sampling, both of which could be very useful. Gaussian has a preference to sample space near blocking obstacles, allowing the agent to reach confined points more easily. Bridge obstacle sampling has a bias towards generating points in narrow or confined areas. Both of these methods could be compared to trails, and used in conjunction with the information gathered.

Further improvements can also be gained by looking at line checking methods. We did not have time to look at these for this report. Ideally, neither sequential nor binary line checking would be used, as the exact geometry of all the objects is known. We can then use this to identify where lines intersect with the objects much faster than either line checking methods would manage. This is an engineering problem still to be solved.

4 Changes to the research plan

The initial plan since the previous RSMG meeting was to have the following completed:

- Interface between Second Life and Java
- Object Map
- Probabilistic road maps being generated
- Navigation using A* route planning
- Trail map and cluster points
 - Trail information collected
 - Used to help probabilistic road map generation
- Gather base line numbers to test any improvement against
- Compare the trail based probabilistic road maps to standard road maps
- Topological map added to segment the space

Of this, the majority of the work has been done. The interface is complete and allows for agents to be written using the CAST framework, connect to Second Life, collect and use the information received. As each additional component is added the framework becomes stronger and more capable of running long term testing. Base line data for a set of “worst case scenarios” has been gathered and can be used for comparison when changing methods of segmentation. The trail information has been gathered and cluster points are able to be generated, though this has not been included in the probabilistic road map generation.

Some preliminary tests showed that while trail points can be used to generate PRMs, how people are moving in the environment affects the usefulness of these. People tend to make very small movements when idling in an area, talking to people or interacting with the environment. This can lead to a very large number of points being generated in a small space. We need to investigate some method of clustering these points together to make them more useful to PRM generation.

Still remaining is work on how to extract a topological representation from the object map and how to segment space. One method which may be very useful for this is spectral clustering (Ng et al., 2001). One use of this is to generate cluster points from trails and use this to generate PRMs. Another potential use is in environment segmentation. By detecting where cluster points exist the agent can use this information to ensure that places that are closely related stay within the same segment.

Trails are useful when they exist, but when they do not the agent still needs to be able to intelligently split the space. We can investigate how we can use machine learning, based on the top down map of the environment and the trail information in other environments to identify potential cluster points and use this to split the environment into smaller segments.

5 Proposed timetable

There is still a substantial amount of work needing to be done on how to segment the space and to generate abstract, topological representations of the environment, as well as the best ways of gathering and using these maps. The remainder of the work we propose to split up as follows.

June - September 2011

Over this time we want to finish the Probabilistic Map comparisons and look at segmentation. By the end of September be able to state what a good map is for the agent and write a paper about our findings. Then, in October, we can look at how to generate these maps on the fly.

- Look at PRM generation in different environments
 - Create some environments that test specific properties of each of the different point generation methods
 - Look at different point generation methods, including Gaussian and bridge obstacle sampling
- Look at how to include trail points and routes in a PRM
 - Look at methods for clustering neighbouring points, such as spectral clustering
 - Test the inclusion of trail points against the other PRM generation methods
- Run all the PRM generation experiments on the cluster
- Look at methods for segmenting space and so include large scale maps
- Use the PRM generation methods for comparing the different methods of segmentation

If there is time we also want to investigate the potential for an agent to learn maps and how to segment space from previous experience.

October - November 2011

Investigate methods of generating the map, and multi-agent techniques. This has been moved back in the schedule to allow for more time focusing on generating maps of large-scale environments.

December - May 2012

Look at using multiple agents to generate maps of large-scale dynamic worlds. The next RSMG report is due in May, so by this point the map representation should be finished and the vast majority of the multi-agent work completed.

May - October 2012

Write thesis.

6 Additions to previous bibliography

“A comparative study of probabilistic road map planners” (Geraerts and Overmars, 2002) was very useful when looking at PRM generation. It described some of the choices that can be made when generating a PRM and what techniques to compare. Halton sequences were looked at in this paper, but found to be no more useful than random point generation. Branicky et al. (2001) go into more depth about quasi-randomised point generation methods. It finds the opposite conclusion to Geraerts and Overmars. It concludes that quasi-randomised sequences, such as halton, are better than purely random methods of selecting points. Another useful resource was Planning Algorithms (LaValle, 2006), chapter four goes into detail about pseudo and quasi randomised points. An extension to PRMs in uncertain environments was covered by Jaillet and Simeon (2004). This introduces the idea of confidence values attached which can be used by the agent to decide if a route is safe to use or not. The experiments were all done in a small-scale, office-like environment. Lavalley (1998) introduces rapidly exploring randomised trees, an alternative to PRMs. This may be useful for comparison, more reading is required on the most recent advances in this area. Missiuro and Roy (2006) look at adapting PRMs to handle uncertain maps. They compare uniform sampling with Gaussian and bridge obstacle based sampling. Gaussian samples the space near blocking objects, and bridge has a bias towards sampling narrow areas. Both of these are quite useful in the type of environments that the agent will be using. Again, a belief or confidence value is placed on the quality of the route.

Mekni and Moulin (2010) look at path planning and map building in a hierarchical way. It makes an interesting claim that other planners are incapable of taking different types of agents into account when planning a route using a single map. They aim to address this issue in large-scale virtual environments by representing the world as a hierarchical map the search space is reduced for route planning. This is similar to the idea of segmenting space and using different levels of the map to plan routes over time. Hahnel et al. (2002) look at methods of building maps in populated areas. This involves classifying fast moving objects as humans, and so removing them from the map entirely.

Cocaud and Jnifene (2010) use probabilistic quad trees to segment space in large-scale environments. By splitting the world into gradually smaller cubes they are able to say whether every portion of the world is free or not. It does not state the underlying map representation used, but this is one method of segmentation which may be useful when applied to the object map. Map updates are cyclical, so the agent can update its map as it moves.

References

- Michael S. Branicky, Steven M. Lavalle, Kari Olson, and Libo Yang. Quasi-randomized path planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, pages 1481–1487, 2001.
- C. Cocaud and A. Jnifene. Environment mapping using probabilistic quadtree for the guidance and control of autonomous mobile robots. In *Autonomous and Intelligent Systems (AIS), 2010 International Conference on*, pages 1–6, june 2010. doi: 10.1109/AIS.2010.5547019.
- Roland Geraerts and Mark H. Overmars. A comparative study of probabilistic roadmap planners. In *Workshop on the algorithmic foundations of robotics*, pages 43–57, 2002.
- D. Hahnel, D. Schulz, and W. Burgard. Map building with mobile robots in populated environments. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, pages 496 – 501 vol.1, 2002. doi: 10.1109/IRDS.2002.1041439.
- L. Jaillet and T. Simeon. A prm-based motion planner for dynamically changing environments. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 2, pages 1606 – 1611 vol.2, sept.-2 oct. 2004. doi: 10.1109/IROS.2004.1389625.
- Lydia E. Kavraki, Jean-Claude Latombe, and E. Latombe. Probabilistic roadmaps for robot path planning, 1998.
- S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at <http://planning.cs.uiuc.edu/>.
- Steven M. Lavalle. Rapidly-exploring random trees: A new tool for path planning. Technical report, 1998.
- M. Mekni and B. Moulin. Hierarchical path planning for multi-agent systems situated in informed virtual geographic environments. In *Information, Process, and Knowledge Management, 2010. eKNOW '10. Second International Conference on*, pages 48 –55, feb. 2010. doi: 10.1109/eKNOW.2010.11.
- Patrycja E Missiuro and Nicholas Roy. Adapting probabilistic roadmaps to handle uncertain maps. *Proceedings 2006 IEEE International Conference on Robotics and Automation 2006 ICRA 2006*, (May):1261–1267, 2006. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1641882>.
- Andrew Y. Ng, Michael I. Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems*, pages 849–856. MIT Press, 2001.
- Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2003. ISBN 0137903952. URL <http://portal.acm.org/citation.cfm?id=773294>.