NARROW PASSAGE IDENTIFICATION USING CELL DECOMPOSITION APPROXIMATION AND MINIMUM SPANNING TREE

Ahmad Abbadi[1,2], Radomil Matousek[1], Lukas Knispel[1]

[1] Brno University of Technology
Institute of Automation and Computer Science
Technicka 2896/2, 616 69 Brno
[2] Masaryk University
Department of Information Technologies
Botanicka 554/68a, 602 00 Brno
Czech Republic
ahmad.abbadi@mail.com, y107798@stud.fme.vutbr.cz, matousek@fme.vutbr.cz

Abstract: *Narrow passage problem is a problematic issue facing the sampling-based motion planner. In this paper, a new approach for narrow areas identification is proposed. The quad-tree cell-decomposition approximation is used to divide the free workspace into smaller cells, and build a graph of adjacency for these. The proposed method follows the graph edges and finds a sequence of cells, which have the same size, preceded and followed by a bigger cell size. The sequence, which has the pattern "bigger-smaller-bigger" cells size, is more likely to be located in a narrow area. The minimum spanning tree algorithm is used, to linearize adjacency graph. Many methods have been proposed to manipulate the edges cost in the graph, in order to make the generated spanning tree traverse through narrow passages in detectable ways. Five methods have been proposed, some of them give bad results, and the others give better on in simulations.*

Keywords: *Narrow passage, cell decomposition, minimum spanning tree MST, Motion planning, sampling based.*

1 Introduction

The narrow passage problem is a common problem for probabilistic planning algorithms, which are used in the motion-planning domain. The problem occurs when a uniform distribution is used. Since, the small volume reduces the sampling probability.

Many researches focus on narrow area identification in order to enhance the efficiency of sample-based planners in narrow workspaces. They try to increase the samplers' ability to take samples from these important areas. For example, the Gaussian sampler locates points near to the obstacles surface. This helps obtaining substantial points in narrow space, based on the idea that the narrow passage exists between obstacles. This method improves the efficiency of planners [1], [2]. But is still has some difficulties to plan a path through narrow passage, where many points near the obstacle boundaries lie far away from narrow passages. In consequence, it does not help the planner to find a solution. Another method has been proposed in [3], to increase samples in narrow areas. The randomized bridge builder (RBB) assumes that the narrow areas appear between obstacles; therefore, it randomly takes two configurations in obstacle space and tests the middle points between them. If a middle point is located in the free space, the algorithm keeps it as a milestone in the narrow area. This process attempts to bridge the gap and generate configurations in a narrow passage. After that, the algorithm takes these middle points as milestones to build PRM. However, there is still recognized present of these samples lying in the corners and hollows of the obstacles [2], [4], [5]. In addition, this method requires a long time to cover the narrow areas. It may fail many times before successfully bridging a gap. Because it needs, a sequence of three nodes such that the endpoints are in obstacle space and the midpoint is in free space. Another research proposed a hybrid sampling strategy, which uses uniform sampler and randomized star builder (RSB) [5]. The RSB is used to identify narrow passages in the workspace. It is an improved version of RBB, and it depends on more than two points lie in obstacles to build the bridge. This method boosts the narrow passage identification. However, it still may fail many times before cover the narrow area. Another algorithm was proposed in [6]. The toggle PRM methodology builds a graph structure for both free and obstacle spaces. These graphs use the information about collision to generate samples, which are used later to generate other samples within the narrow passage. If a sample is chosen randomly from free space, it is added to the free graph. The PRM tries to connect this free sample to the nearest nodes in the graph. If a collision with obstacles happened during this connection, the collision points are stored as nodes in the obstacle graph. Later on, the graphs are toggled, and the PRM tries to connect the nearest nodes of the obstacle graph to the generated collision points. During this connection, another point could be generated in the free area because of collision with free space, and repeatedly, this point will generate other samples and so on. This method is efficient, but many samples are generated near the corners and hollows. Other works, try to avoid the narrow passage effects on the planning algorithms using hybrid approaches [7].

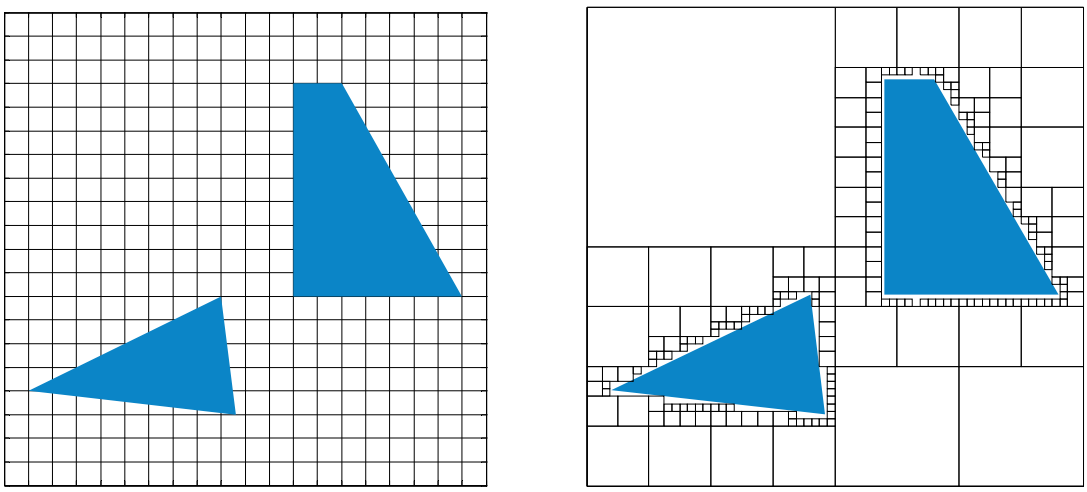
Our proposal for narrow-passage identification uses approximate cell decomposition (ACD) and minimum spanning tree (MST) algorithms. The former one utilized to divide the free space into manageable cells and define the adjacency relation between free areas explicitly using a graph of adjacency. The MST is used to remove the cycled connections in that graph. In the next section 2, the ACD approaches are reviewed, and then the proposed methods have been presented in section 3. We test our proposal in two workspaces and the results is shown and discussed in section 4. Finally, we conclude the paper in section 5.

2 Cell decomposition approximation

Cell decomposition algorithms (CDAs) are one of the old applicable solutions in motion planning problem. These methods find the obstacles and the free regions, and build a graph of adjacency for the free ones [8, Ch. 6], [9]. In general, the cell decomposition algorithms are classified into two categories; the exact cell decomposition methods and the approximation methods [10]–[12]. The first category uses geometric algorithms to determine the free areas and build free cells explicitly [13], [14], where the union of all generated cells is equal to the free space exactly. However, finding exact free cells is not an easy task, especially in higher dimension spaces. That leads to the second category, which uses the approximation techniques to divide the workspace, e.g. quad-tree, octree division, and voxel grid, etc., [15], [16].

The CDAs divide the workspace into smaller regions called cells, then, they build a connectivity graph according to the adjacency relationships between the free cells. The graph's nodes represent the cells, while the graph's edges represent the adjacency relations between these cells. From this connectivity graph, a continuous path can be found by following the adjacent free cells. The simplest method of approximation is the voxel grid approach. It uses a regular voxel grid, as shown in Fig. 1-a. It excludes the cells on the obstacle areas and builds a graph of adjacency for the cells in the free area. This method is efficient for low dimensional spaces. However, it generates a large number of cells. This method is resolution complete; which means the algorithm's completeness depends on the grid's fine [15], [16].

Quad-tree decomposition is another improvement for cell decomposition approximation. This approach uses a recursive method to divide the workspace. It recursively subdivides the cells until one of the following conditions is satisfied. 1- Each cell lies completely either in a free space or in an obstacle region. 2- Or, an arbitrary limit of a resolution is reached. Once a cell fulfills one of these criteria, it stops decomposing [16, Ch. 14], [17]. Fig. 1-b, shows the generated cells of the quad-tree method.



(a) (b)

Figure 1: Cell decomposition approximation. a: voxel approximation methods. b: quad-tree approximation methods

3 Proposed methods

Narrow passage problem faces most of sampling based approaches. The problem occurs when a uniform distribution is used to take samples form the workspace, because the small and narrow areas have a low probability to get samples within their space.

In this work, the quad-tree cell decomposition approximation is used to find a graph of adjacency for the free cells, where the graph nodes represent the free cells and the edges represent the connection relations between these free cells.

We exploit the information about the cells size to find the narrow area. Our proposal based on the idea of following the adjacent cells size. If the translation is done from a big cell to others smaller ones, which have the same size, then followed by a translation to another bigger cell, then this sequence of the small same size cells is most likely to be a narrow passage or important area from motion planning point of view. Fig. 2 shows an example, where the shaded cells represent the most important region in this workspace.

To implement the proposed method, a preprocessing step should be applied to the adjacency graph. Since, the graph of adjacency has many loops and cycled connections between the nodes, for that, a linearization of the graph should be done before the narrow passage identification method is applied. Based on that, the MST approach is used to build a new liner graph.

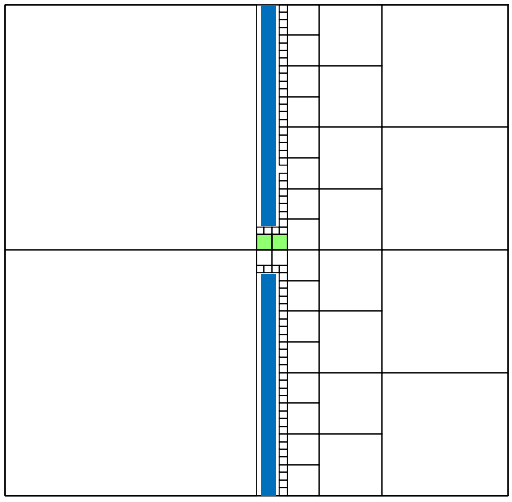


Figure 2: Example of the narrow passage identification

The MST tries to build a spanning tree that has the lowest cost, and contains all nodes visited one time. This principle causes another problem, where the tree is planned in unpredictable regions in the workspace based on the edges costs. In order to solve this problem, the edges weights, which affect the spanning tree construction process, is updated and adapted. The weights are manipulated, in order to give a low cost for edges that placed within narrow and small areas, and at the same time, prevent the MST method of constructing the tree structure near to the obstacles boundaries. Many ideas for weights manipulation are tested to generate the desire spanning-tree. We test five methods.

The first method uses the real distance between cells.

The second one uses the uniform cost for translating from one cell to another one. This method based on the idea that, the generated tree should minimize the path cost by using the minimal number of translations; in consequence use the bigger cells when it is possible.

The third proposed method, the bias to different cells size, updates the edges' weight in such a way that it makes the cost of translation between different cells' size lower than translation between cells that have the same size. This method makes the span tree uses the smaller cell as leaves while it uses the bigger cell as roots.

The fourth method, the bias to equal cells size, suggests giving the lowest cost to the translation between the same size cells. It is the opposite of the previous method, the idea behind this proposal is to make the cells that have the same size a sequence, does not satisfy the narrow passage condition "bigger-smaller-bigger," and the MST will construct this pattern just when it is necessary.

The last proposed method, the disproportional cost to the distance, gives the edges costs based on cells size, the smallest cost is given to translating between the bigger cells. We realize this proposal by finding the longest distance between cells then subtracts all translation distances from that distance. The result is given as a weight of the graph edges. This method gives the translation between the largest cells, which have the longest distance, the lowest cost, while the translation between smaller cells will have higher costs.

4 Results and discussion

The proposed methods are simulated and tested in two workspaces. The first one is an office-like workspace, where there is one route to connect any two rooms. The second workspace is generated in such a way that the connections between the free regions have multi-routes.

The result is shown graphically using grading colors, where each color represents a narrow passage sequence. The size of the shaded sequence represents the size of the corresponding cells.

The results of the first and second methods show that the algorithm finds many narrow passages, which are in these scenarios represent the areas between rooms and large free areas. But the results are considered failed because it generates many sequences near the obstacles and far away from the narrow passage.

The first method that uses the real distance as a cost, makes the MST constructs the tree near the obstacle and follows the smaller cells as shown in Fig. 3. Where the left figures (a,c) show the ACD and MST graph graphically while the right figures (b,d) show the sequences of the narrow passages, which are presented using multi-level of colors.

The uniform cost method generates a tree structure which uses more bigger cells as expected, and it generates a better solution, however the result still not good and unreliable. Fig. 4-a,c show the generated spanning tree in both workspaces, while the narrow passages sequences are shown in Fig. 4-b,d.

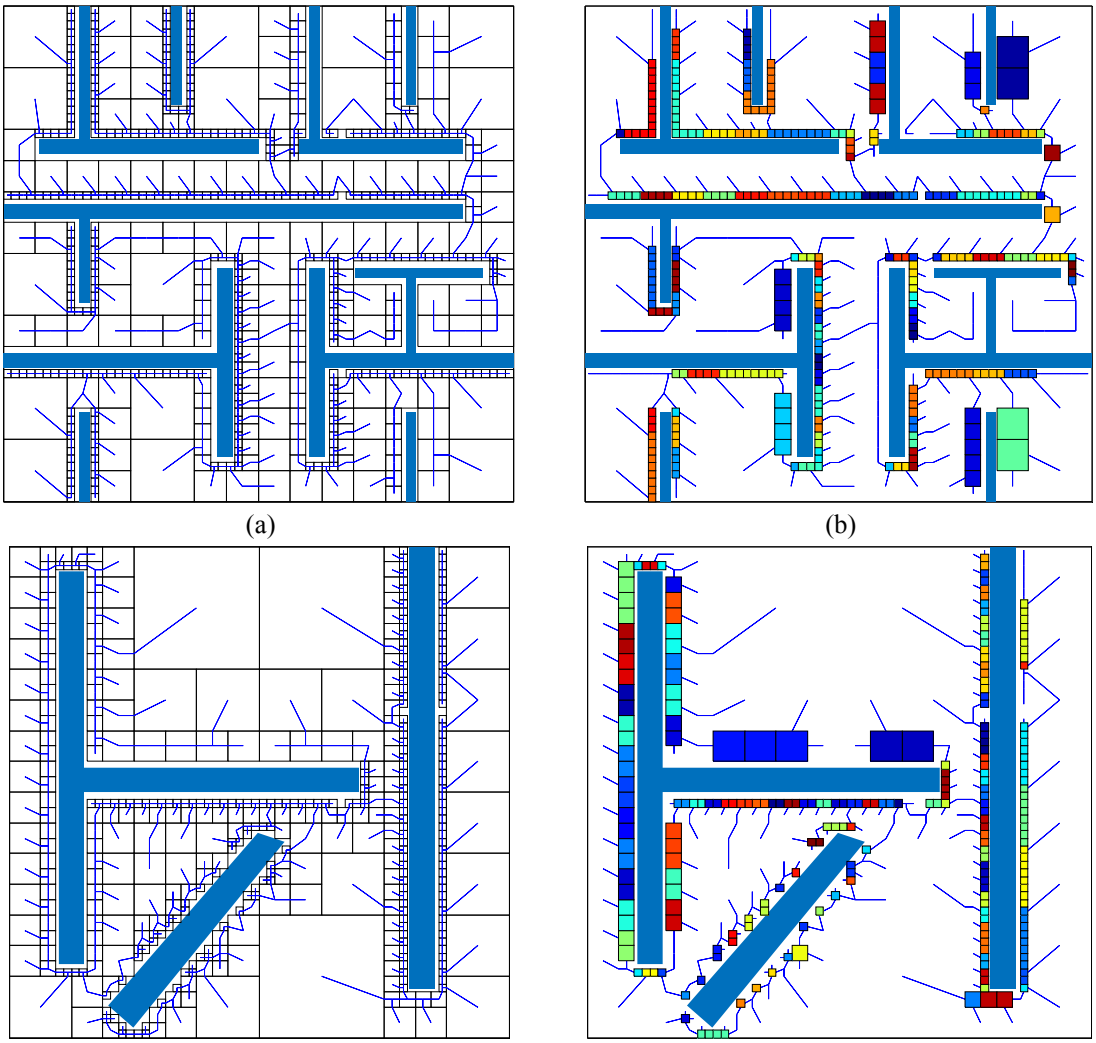
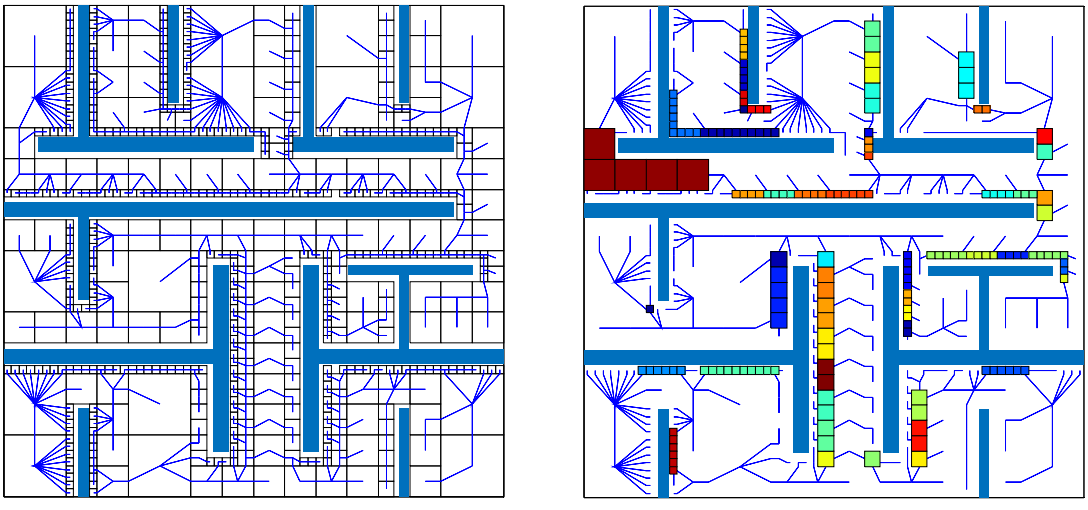
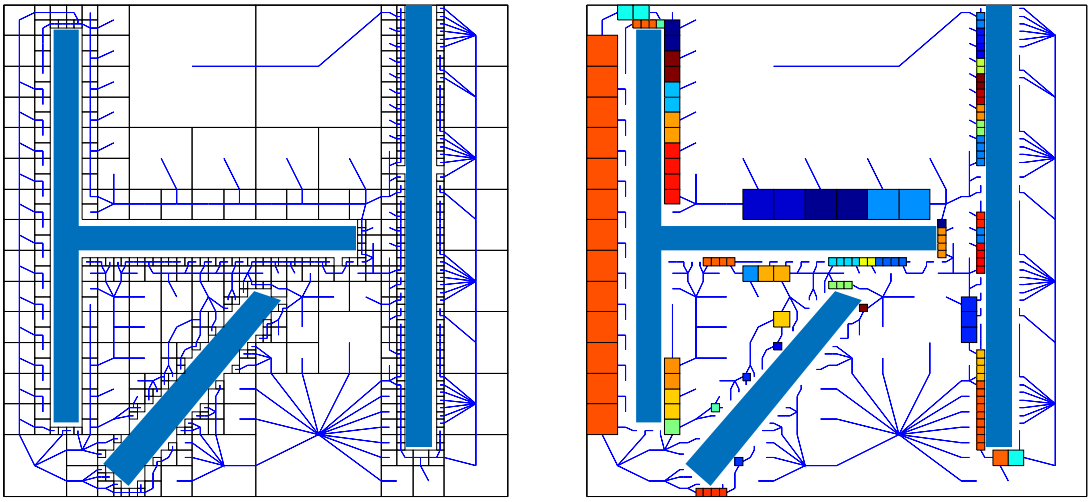


Figure 3: Edges cost equal to the real distance



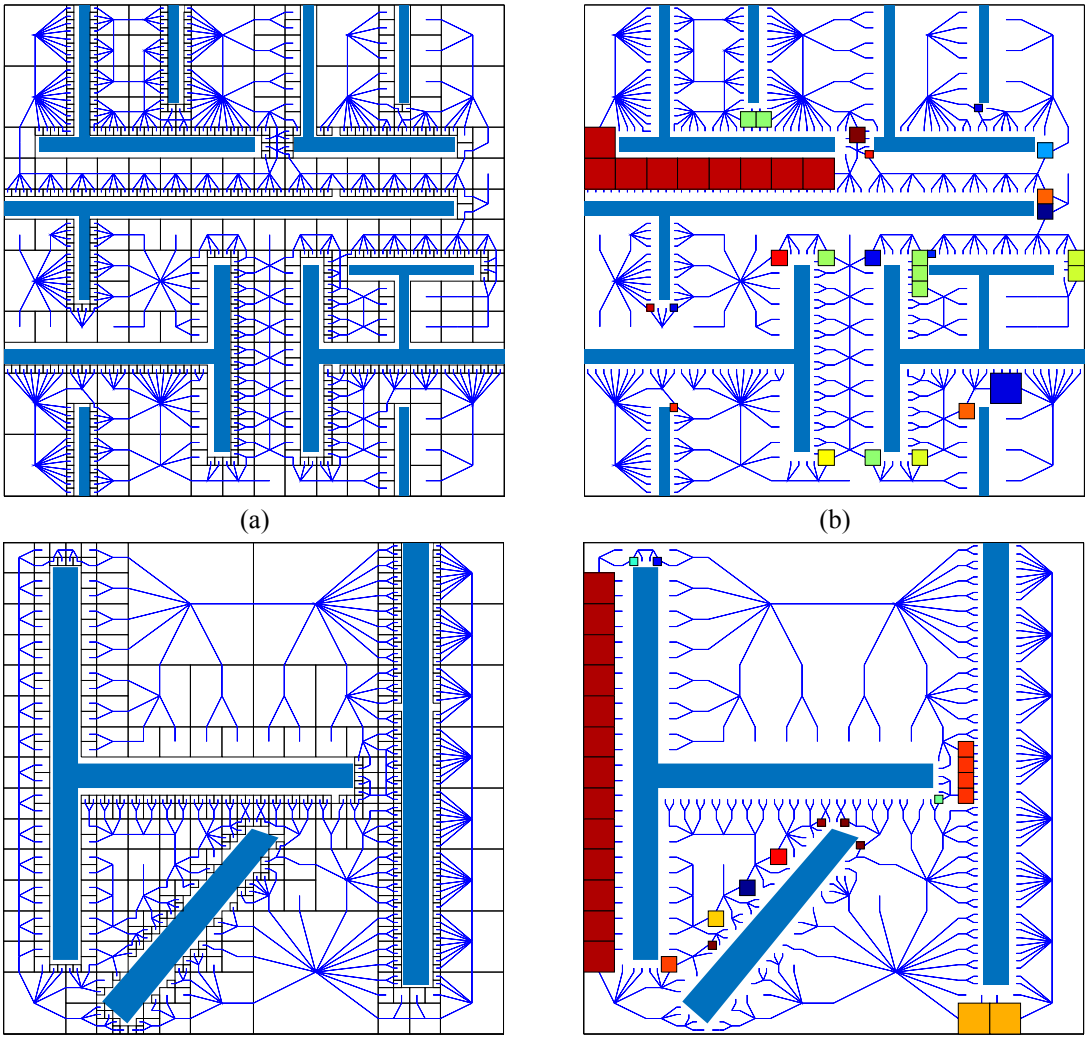


(c) (d)

Figure 4: Uniform translation cost

The third method directs the MST algorithm to use different cell size. The generated trees translate between cells that have different size more than the translation between the cells that have the same size, Fig. 5-a,c show the spanning trees.

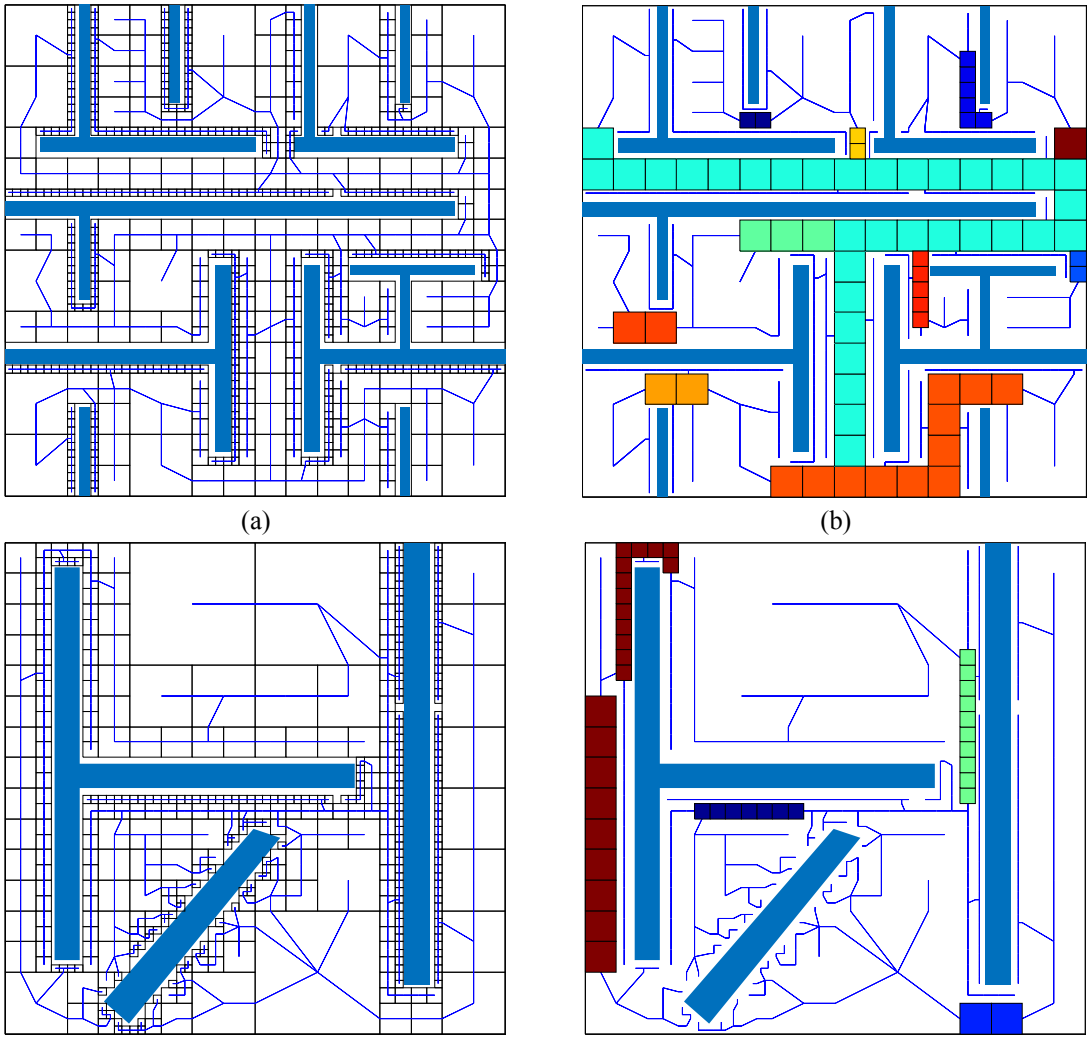
This method generates a better solution as shown in Fig. 5-b,d. However, it also generates long sequences and undesired sequences, especially on the second workspace, which has un-alignment obstacle to the axis Fig. 5-d.



(a) (b)

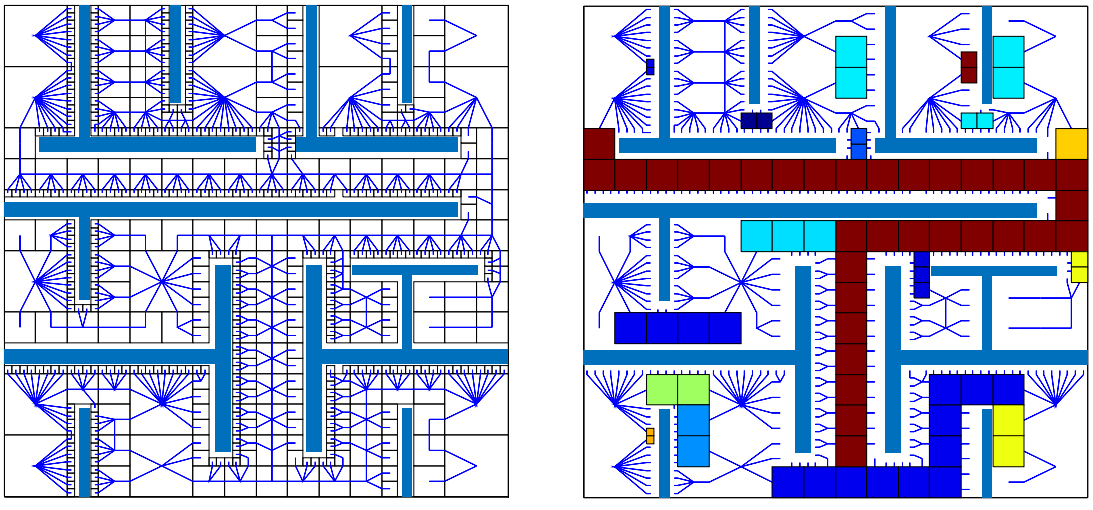
(c) (d)

Figure 5: Bias to different cells size

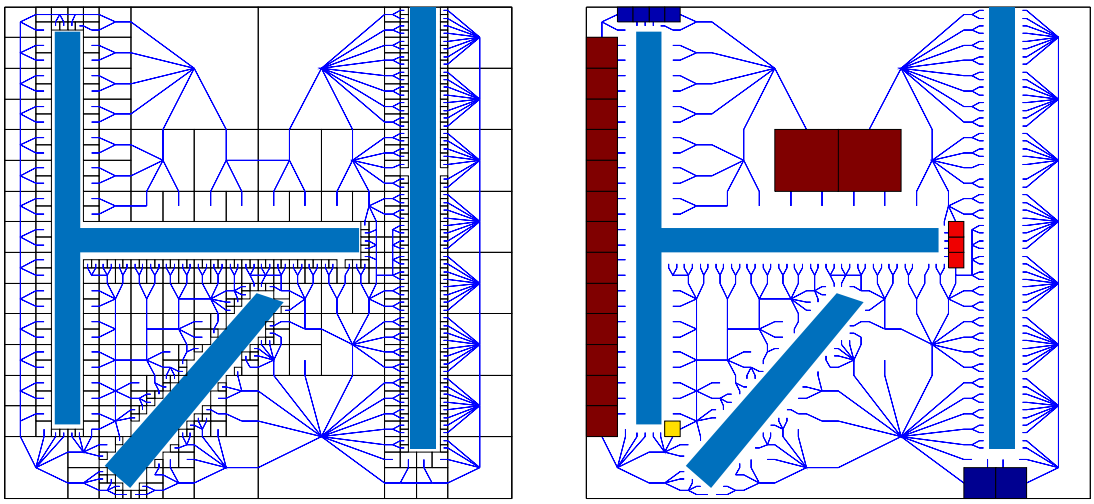


(a) (b) (c) (d)

Figure 6: Bias to equal cells size



(a) (b)



(c) (d)

Figure 7: Disproportional cost to the distance

The fourth method, which gives lower cost to the translation between equal cells size as shown in Fig. 6-a,c, generates better results, it has the ability to find all narrow passage. But, it generates very long sequences as shown in Fig. 6-b,d.

The last proposed method, which has disproportional cost to the distance, generates spanning trees as shown in Fig. 7-a,c. It produces a relatively good solution. However, it is still has a problem with sequences generation, since it has some faults to finds the narrow passage, in addition the generated sequences are long, and sometime they merge many narrow passages together as shown in Fig. 7-b,d.

5 Conclusion

In this work, the narrow passage identification problem is discussed. The narrow areas are a problematic issue faces the sampling-based motion planner. The cell-decomposition approximation algorithm is utilized to find the free regions and build a graph of adjacency for them based on the adjacency information.

The proposed method to identify the narrow passage, finds a sequence of cells along the graph edges that have the same size, preceding and followed by a bigger cells size. This smaller sequence is more likely to be located in the narrow passage.

Because of the graph of adjacency characteristic, which has many loop connections between the adjacent cells the minimal spanning tree algorithm is used to linearize this graph. Many methods have been proposed to manipulate the edges cost in the graph, in order to make the generated spanning tree traverse through narrow passages in a detectable way, which means following the pattern of the narrow area “bigger-smaller-bigger” sequence of cells. Five methods are proposed, some of them give bad results and the others give better results as shown in the simulation.

More studies and analysis to the cost manipulation process should be reviewed in the future work, in addition, the bad method to identify the narrow passage can be updated to find obstacles boundaries cells, based on that, the non-uniform distribution can be introduced to be used in the motion planning samplers.

Acknowledgement: This work was partially supported by grant of BUT IGA No. FSI-S-14-2533: "*Applied Computer Science and Control*".

References

[1] Boor, V., Overmars, M.H., van der Stappen, A.F.: The Gaussian sampling strategy for probabilistic roadmap planners, in *1999 IEEE International Conference on Robotics and Automation, 1999. Proceedings*, vol. 2, pp. 1018–1023, (1999)

[2] Lin, Y.-T.: The Gaussian PRM Sampling for Dynamic Configuration Spaces, in *9th International Conference on Control, Automation, Robotics and Vision, ICARCV ’06*, pp. 1–5. (2006)

[3] Hsu, D., Jiang, T., Reif, J., Sun, Z.: The bridge test for sampling narrow passages with probabilistic roadmap planners, in *IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA ’03*, vol. 3, pp. 4420–4426. (2003)

[4] Sun, Z., Hsu, D., Jiang, T., Kurniawati, H., Reif, J.H.: Narrow passage sampling for probabilistic roadmap planning, *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1105–1115, Dec. (2005)