VIZMO++: a Visualization, Authoring, and Educational Tool for Motion Planning*

Aimée Vargas Estrada, Jyh-Ming Lien, Nancy M. Amato

Parasol Laboratory, Department of Computer Science
Texas A&M University
College Station, Texas, 77843-3112, USA
{*aimee,neilien,amato*}*@cs.tamu.edu*

Abstract— **Comprehension of concepts and algorithms involved in the robotics field can be improved through the use of an interactive visualization tool. In this paper we present an interactive tool for visualizing and editing motion planning environments, problem instances, and their solutions. Teachers can take advantage of visualization tools to help their students to better understand motion planning and its complexity as well as the different strategies that have been developed to solve the motion planning problem. While the tool we present allows the animation, manipulation, and evaluation of solution paths found by any motion planner, it is specialized for sampling-based randomized planners such as Probabilistic Roadmap (PRM) and Rapidly-Exploring Random Tree (RRT) methods.**

I. INTRODUCTION

Motion planning is the problem of finding feasible paths for movable objects among obstacles [1]. There are many fundamental concepts in motion planning, such as degrees of freedom, configurations, and configuration space, that can be difficult concepts to understand. In the classroom, a visualization tool that illustrates a motion planning problem, such as the shape of a robot, the environment in which the robot is operating, start and goal configurations, and valid paths for the robot, can help students to more easily understand these concepts. A good visualization tool for motion planning can also provide students access to workspaces that they are otherwise not able to see. Examples of this include manufacturing plants, an airplane engine or outer space.

In addition, it is known that one of the easiest ways for students to learn an algorithm is through the animation of the algorithm. In [2], Brown and Sedgewick presented an interactive system for this purpose. Similarly, a visualization tool that can display the outcome of different motion planning strategies can be a very helpful resource to understand and compare these strategies. For example, with a visualization tool students can easily distinguish the differences between the results from an obstacle-based planner [3], which generates feasible configurations around obstacles, and the results from a Medial-Axis-based motion planner [4], [5], which generates feasible configurations on or near the Medial Axis of the feasible space. Moreover, abstract concepts, such as potential field, roadmap, tree, connected components, and closest neighbors,

that are commonly used in motion planning methods, can be understood more easily using the illustrations provided by a visualization tool.

Another good way to improve students' understanding of motion planning is to allow them to modify existing problems, play with the parameters of a motion planning strategy, or to create new problem instances. Via these processes, students can see how the changes to the workspaces and the parameters can affect the outcome of various motion planning methods. They can see the solutions of the problems and they can learn to anticipate and speculate on the possible outcomes for the problems that they have created.

It is for these reasons that we created our visualization tool, VIZMO++, which offers multiple means to visualize the robot, the obstacles, and planner outputs (e.g., paths and roadmap) in a 3D environment. With this tool students can observe and gain better understanding of the instances of the motion planning problem and the computations of the motion planning methods, thus better understanding the concepts involved.

Besides visualization, our tool provides a means to edit existing problems and create new problem instances. VIZMO++ allows users to change the properties of the objects in the scene such as the position or orientation of the obstacles and the robot. Obstacles can be added or deleted from the robot's workspace and VIZMO++ allows easy demonstration of the results of performing such operations, e.g., recomputing the roadmap. Students can also experiment and explore, outside of the classroom, with the functionality VIZMO++ offers.

VIZMO++ has already been used by Professor Lucia K. Dale at the University of the South [6]. Students, from seniors to freshman, found it easy to use especially in creating new environments for various experiments. VIZMO++ also helped them understand the different algorithms for roadmap generation.

II. RELATED WORK

In this section, we discuss related work on visualization tools for motion planning and methods that have been proposed for motion planning problems.

A. Visualization tools for motion planning

Some research groups have developed visualization tools to support motion planning. Among those tools is the Motion Strategy Library (MSL) [7]. It was developed by the research

group of Professor Steven M. LaValle at the University of Illinois at Urbana Champaign. MSL is an open source and free software to develop and test motion planning algorithms. MSL also allows users to generate a two-dimensional plot of the projection of the connectivity graph on the plane formed by any two variables selected by the user. With MSL users can see an animation of the path found.

B. Motion Planning Methods

The motion planning problem consists of finding a valid path between a start and a goal configuration for a movable object or robot [1]. A configuration is defined as an *n*-tuple of values that determine the position of every point of the object. The workspace is the place in which the robot moves. In traditional robotics and animation applications, the workspace is composed of one or more objects (called obstacles). The configuration space (C-space) [8] is an *n*-dimensional space (*n* being the number of degrees of freedom (DOF) of the robot) consisting of all (i.e., feasible and infeasible) robot configurations. The robot is represented as a point in C-space.

The motion planning problem is known to be PSPACE-hard [9], at least as hard as an NP-complete problem. All complete algorithms developed so far take exponential time in the number of DOF of the robot. A practical and broadly used approach to solve the motion planning problem is through randomization. Randomized algorithms have been useful in finding approximate solutions to intractable problems such as motion planning. In these methods the movable object's C-space is sampled at random and the samples are connected to form collections of feasible paths in C-space. We can classify most randomized planners as either roadmap-based [3], [4], [10] or tree-based [11]–[13]. Notable examples of roadmap-based methods are the Probabilistic Roadmap Methods (PRMs) [3], [4], [10], [14]–[18], and of tree-based planners are Ariadne's Clew algorithm [12] and the Rapidly-Exploring Random Tree (RRT) [13].

III. VIZMO++ : AN OVERVIEW

VIZMO++ is a 3D visualization/authoring tool conceived to display and manipulate the elements in the robot's workspace and the information generated from motion planners (roadmap and path), but specialized for sampling-based (graph and tree) motion planners. Figure 1 shows VIZMO++'s graphical user interface (GUI) and the U-tunnel environment which is composed of a U-shaped tunnel that the robot has to pass through in order to reach its goal.

VIZMO++ is also an educational tool that can help students in their understanding. For example, students can learn and understand different planner strategies and solutions through straightforward visualization, animation and manipulation. Since different planners generate different sample distributions of configurations in C-space, roadmap visualization can help students to better understand how each method works by looking at the distribution of the nodes and the connectivity of the roadmap.
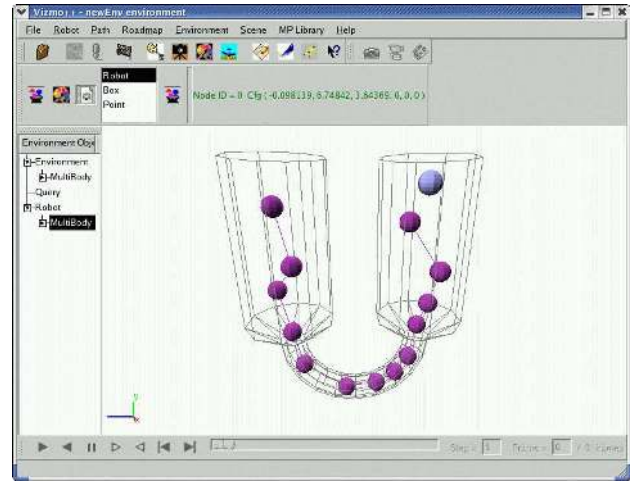

Fig. 1. A snapshot of VIZMO++. A roadmap is shown in a workspace consisting of two chambers connnected by a U-tunnel.

Using VIZMO++, students can also modify or create environments, e.g., adding or deleting obstacles, changing their position or orientation, and running a planner to compute a new roadmap for the new environment. Students can also set new queries, save environments to be able to work on them later, or select, move and/or add nodes and thus create new roadmaps. Our tool also supports visualization of collision between objects, and offers an interface to our motion planning library.

The major components of VIZMO++ include:

- A user friendly interface.
- Visualization of the workspace/scene.
- Visualization of results generated by motion planners.
- Editor for the workspace/scene.
- Editor for roadmaps.

In the following sections, we will present the functionality of VIZMO++ and then discuss three case studies including:

- Creation of motion planning problems.
- Visualization of unusual workspaces.
- Comparison of different planning strategies.

Additional details about VIZMO++'s design and examples of its use can be found in [19].

IV. FUNCTIONALITY OF VIZMO++

A. User interface

The organization and distribution of the elements of the GUI are as follows (see Figure 1):

- The scene window placed in the center of VIZMO++ renders motion planning environments and results produced by motion planning methods. All other GUI components are placed around this window to control what and how objects should be rendered.
- The main menu bar encloses the functions for controlling the elements we are interested in visualizing, that is robot, roadmap, the environment, and the path. The main menu

bar also includes options for opening and saving files, access to our motion planning library, auxiliary tools, and help.

- The toolbar placed at the top contains the most used functions so users can access them quickly including options for rendering and manipulating environments and roadmaps.
- An outline window on the left outlines all the objects that are present in the scene providing a quick access to these objects.
- An animation toolbar placed at the bottom gives access to the animation of the path.

B. Visualizing the scene

When VIZMO++ loads a workspace, students will see the robot, the obstacles, the bounding box of the workspace, and the start and goal configurations of the robot. Students can rotate, zoom in and zoom out to get a better view of the scene.

For most objects in the scene, VIZMO++ supports two rendering modes, solid and wire, that allow students to better understand and explore the environment. For example, some environments can be completely surrounded by walls and so the interesting characteristics of the environment can be hidden. Figure 2 shows these two rendering modes for a *maze* environment. The environment is composed of a series of tunnels. The robot, which is a rigid object, needs to go through tunnels to reach its goal at the bottom of the maze. To change from one state to another, the student only needs to select one or more obstacles and select a rendering mode.

In addition to these workspace objects, VIZMO++ also shows the query in wire mode giving a different color for the start and goal configurations. Users can distinguish them not only by color but also by the labels: 'S' for start and 'G' for goal.

C. Visualizing results

The results generated by a motion planning strategy include a graph (roadmap) and a path. In VIZMO++, a roadmap is colored according to its connected components. Roadmap nodes can be shown as points, boxes, and robot configurations. Point representation shows the location of configurations and is the fastest to render. Box representation shows the location and orientation of configurations and is a bit slower but still quite fast to render. Robot representation shows the actual configuration in the environment but its rendering can be slow. The nodes can be scaled to have clearer visualization when the size of a roadmap is large.

In Figure 3 we show the *walls* environment for a 9-DOF robot. The objective of the robot is to go from the first chamber to the second one and it has to go through a narrow passage, which is a hole on the second wall. In Figure 3(a) we show the roadmap with the nodes rendered as robot configurations. The connected components are rendered in different colors so the student can distinguish them. Students can select connected components that will be highlighted once selected and the number of nodes and edges of these selected components will
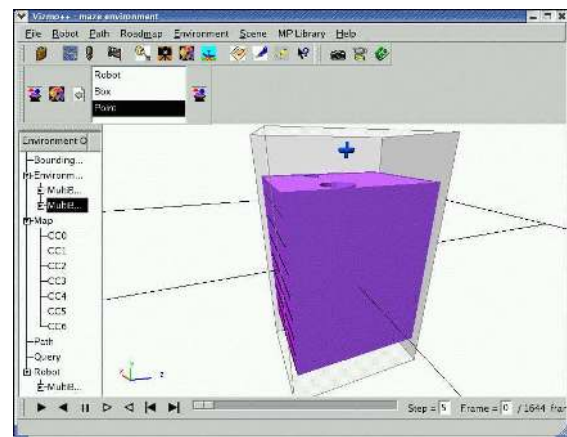
(a)
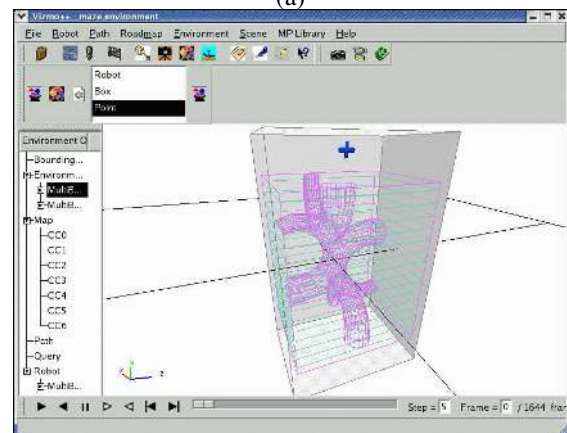
(b)

Fig. 2. The Maze Environment: (a) solid model and (b) wire-frame model.

be displayed on the main window. In Figure 3(b) we show the swept volume of the path. The path can be animated using the *animation* toolbar. The student can play the path or can go frame by frame to see the robot's configuration in more detail.

D. Editing the scene

VIZMO++ provides tools to edit a given scene. For example, obstacles can be selected to change their position and orientation using the *rotation* and *translation* tools. Figure 4(a) shows how an obstacle is moved using the translation tool. The collision detection option can be turned on so users can know whether the obstacle is in collision with the robot. VIZMO++ notifies the user by changing the robot's color. Obstacles can also be added or deleted from the scene or have their color changed. All these options allow the student to create their own environments. Once an environment has been changed, the student can recompute the roadmap for the new environment by just clicking on the *automatic roadmap* button and visualize the result.

E. Editing roadmaps

VIZMO++ provides users with tools to generate a new roadmap by hand, to modify an existing roadmap or to add
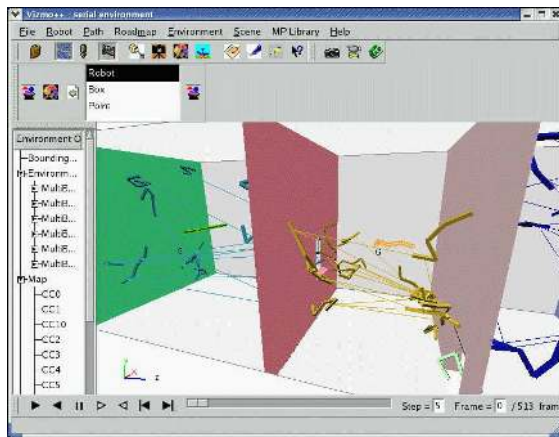
(a)
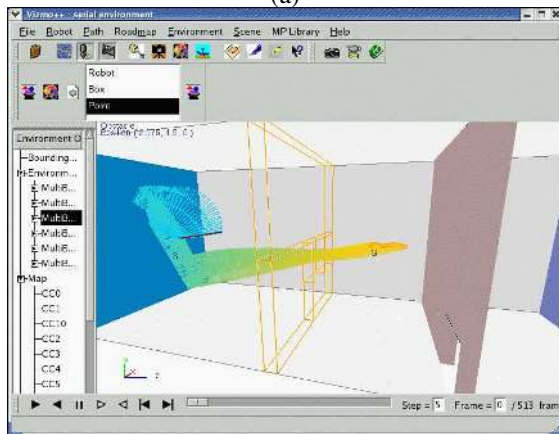
(b)

Fig. 3. *walls* environment. (a) The roadmap and the nodes rendered as actual robot configurations. (b) The swept volume of the path.

(a)

(b)

Fig. 4. (a) The *translation* tool. An obstacle is moved and users can activate the collision detection option so obstacles will not be placed in collision with the robot. (b) The *rotation* tool. The robot is selected to change its orientation.

nodes to it.

A node of a given roadmap can be selected to either see or modify its configuration. VIZMO++ also enables users to copy existing nodes, add new nodes to the roadmap, and modify a configuration and visualize how that configuration changes accordingly. Figure 5(b) shows the GUI to add a node and change its DOF through a dialog box. Users can also add edges to the roadmap by selecting a pair of nodes, however, the current version of VIZMO++ does not check whether the edge is valid or not.
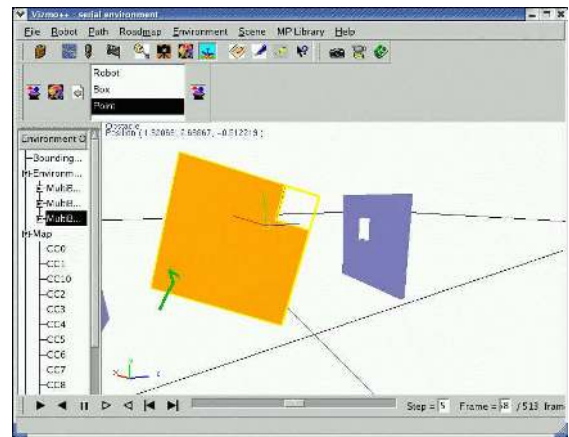
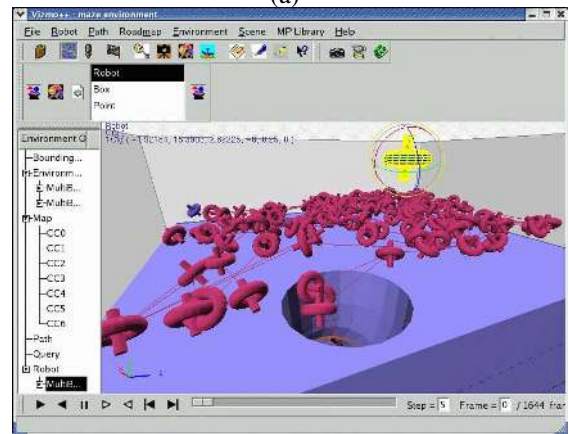## V. CASE STUDIES

A. Creating motion planning problems

Creating a new environment. VIZMO++ can be used to create new environments. Once an empty environment has been loaded, users can perform the following operations: add or delete obstacles or move and rotate obstacles to modify or create environments.

A new environment can be created by following these steps:

1) Obstacles are added using the 'add obstacle' option from the Environment menu. A dialog window will appear and the user can navigate through the file system to find the file to include. After selecting the file, a confirmation message will pop-up and then the new obstacle will be added to the environment and the user can place it in the desired position.

2) The user might want to change the color of the obstacles and can do it by selecting the obstacle and selecting the color option by right-clicking on the obstacle and selecting the option from the context menu that will appear.

3) More obstacles can be added by repeating the process described above. Once the user is happy with the new environment, it can be saved to a file.

Creating a new query. In VIZMO++, the robot's position can be altered thus giving the possibility of creating new queries. The user can follow the following steps to set a new query.

1) First the robot is selected and then the *set query* option from the *Path* menu is selected. The user has to specify if the configuration will be a start or a goal.

2) After selecting this option, a window will pop-up and will show all the values for the DOFs of the robot, which can be modified in the window or by using the
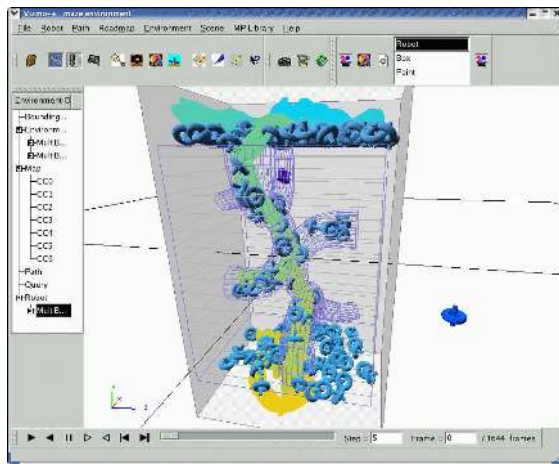
(a)
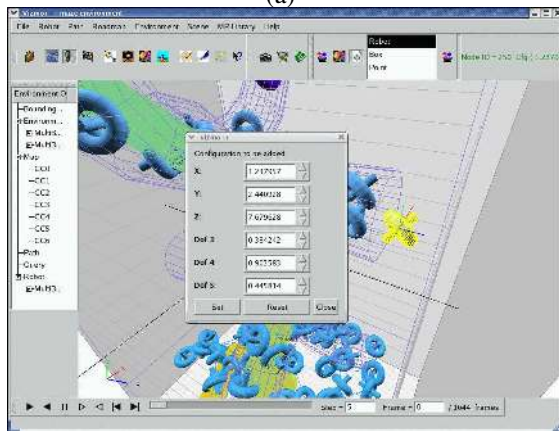
(b)

Fig. 5.   (a) The swept volume of the path and the roadmap for the *maze* environment. (b) The GUI to add a node to the roadmap for the same environment. All the DOF can be changed and students can see how the configuration is updated in the VIZMO++ main window.

rotation and translation tools described before. (Figure 4 (b) shows the robot being rotated). As values change, the user will see how the robot changes its configuration accordingly.

3) After setting start and goal configurations, the user has the option of saving them into a file or computing a path for the new query by clicking on the *run query* button so a new path will be computed and shown if found.

B. Visualization of unusual workspaces

VIZMO++ can also provide students access to workspaces that they are otherwise not able to see. Examples of this include manufacturing plants and mechanical designs, such as an aircraft engine, and workspaces that they are not likely to explore without expensive equipment, such as outer space. This can even include workspaces that are impossible to visit, such as a prehistoric city.

In Figure 6 we show two environments. The first environment is an airplane engine model which contains very complex components. The second environment contains a robot with a complex shape that represents a protein model.
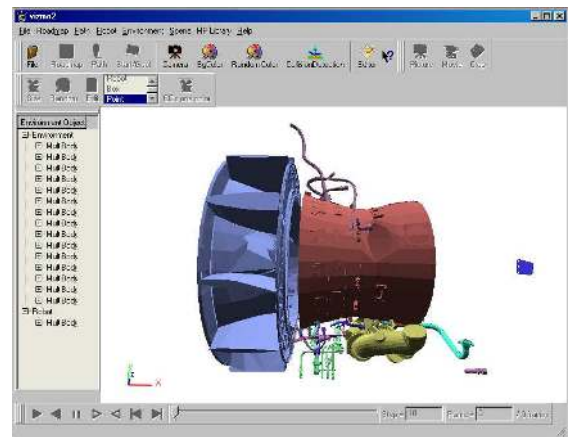
(a)
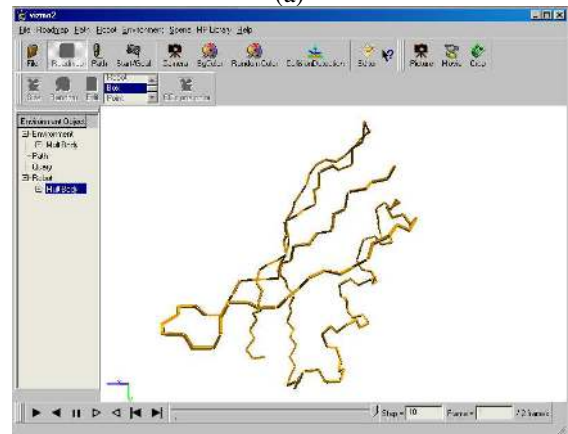
(b)

Fig. 6.   Examples of workspaces that are difficult to access. (a) An airplane engine, and (b) a protein.

C. Comparing different planning strategies

VIZMO++ can be used to help students better understand and compare different planning strategies. Figure 7 shows the *rigid* environment which is composed of two plates. The robot's initial configuration is in between the plates and its goal is to get out of them. We show in Figure 7 (a) the node distribution computed using the Bridge Test strategy [17] which generates samples between the plates and in Figure 7 (b) the roadmap computed using the Gaussian strategy [18] which generates samples around obstacles.

VI. CONCLUSIONS AND FUTURE WORK

We presented VIZMO++, a tool for visualizing and editing motion planning environments, problem instances, and their solutions. VIZMO++ offers a self-explanatory graphical user interface that provides functionality that can help students understand robotics and motion planning concepts.

We are convinced that teachers and students can take advantage of visualization tools such as VIZMO++ which in addition to visualization, allows interaction with the robot's workspace. These kind of tools can help students who are not familiar with motion planning concepts or randomized
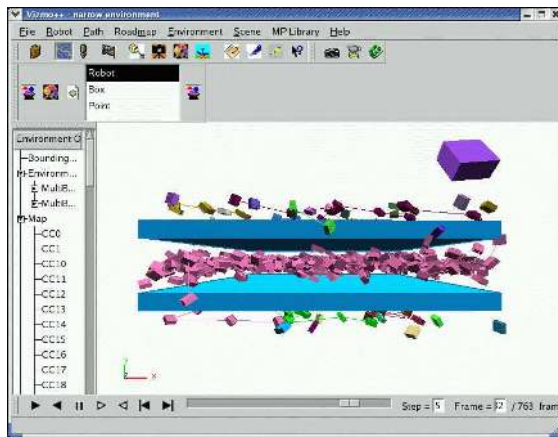
(a)
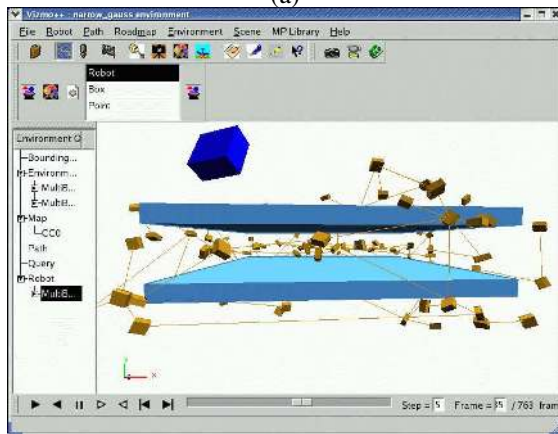
(b)

Fig. 7. For the *plates* environment, (a) a roadmap generated with the Bridge Test strategy and (b) a roadmap generated with the Gaussian Sampling strategy.

motion planning methods, to understand the problem, the difficulty of a given motion planning problem, and show them how different approaches work to solve the same problem while getting them more involved during class with visual demonstrations.

VIZMO++ has already been used for students who were introduced for the first time to the robotics field. While demonstrating to them robotics concepts and motion planning strategies, VIZMO++ was used to illustrate those concepts. It was noted that students showed more interest when they were able to actually see what they were being taught.

Although VIZMO++ offers many functions that can be of help to students to better understand motion planning concepts, there are other functions we need to improve and work on. It can happen that a given problem may deal with more than one robot at the same time. Our current version of VIZMO++ supports just one robot so we will need to extend our current implementation to make it support more than one robot. While a roadmap node is being moved, we want to check for collision to alert the students if they are selecting an invalid configuration or if the new node is outside the bounding box. Currently VIZMO++ does not support either of

these functions. Edges can be added to the roadmap but edges are not checked for validity. To avoid confusion and help even more in the understanding of robot motion planning, we can give a student the option of visualizing just the nodes that were actually used to compute the path.

REFERENCES

[1] J.-C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers, 1991.
[2] M. H. Brown and R. Sedgewick, "A system for algorithm animation," in *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press, 1984, pp. 177–186.
[3] N. M. Amato, O. B. Bayazit, L. K. Dale, C. V. Jones, and D. Vallejo, "OBPRM: An obstacle-based PRM for 3D workspaces," in *Robotics: The Algorithmic Perspective*. Natick, MA: A.K. Peters, 1998, pp. 155–168, proceedings of the Third Workshop on the Algorithmic Foundations of Robotics (WAFR), Houston, TX, 1998.
[4] S. A. Wilmarth, N. M. Amato, and P. F. Stiller, "MAPRM: A probabilistic roadmap planner with sampling on the medial axis of the free space," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2, 1999, pp. 1024–1031.
[5] J.-M. Lien, S. L. Thomas, and N. M. Amato, "A general framework for sampling on the medial axis of the free space," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, September 2003, pp. 4439–4444.
[6] L. K. Dale, January 2006, Personal Communication.
[7] S. M. LaValle, "Motion Strategy Library," http://msl.cs.uiuc.edu/msl/.
[8] T. Lozano-Pérez and M. A. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Communications of the ACM*, vol. 22, no. 10, pp. 560–570, October 1979.
[9] J. H. Reif, "Complexity of the mover's problem and generalizations," in *Proc. IEEE Symp. Foundations of Computer Science (FOCS)*, San Juan, Puerto Rico, October 1979, pp. 421–427.
[10] L. E. Kavraki, P. Svestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, August 1996.
[11] J. Barraquand and J. C. Latombe, "Robot motion planning: A distributed representation approach," *Int. J. Robot. Res.*, vol. 10, no. 6, pp. 628–649, 1991.
[12] P. Bessiere, J. M. Ahuactzin, E. G. Talbi, and E. Mazer, "The Ariadne's clew algorithm: Global planning with local methods," in *Proc. IEEE Int. Conf. Intel. Rob. Syst. (IROS)*, vol. 2, 1993, pp. 1373–1380.
[13] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1999, pp. 473–479.
[14] R. Bohlin and L. E. Kavraki, "Path planning using Lazy PRM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2000, pp. 521–528.
[15] C. L. Nielsen and L. E. Kavraki, "A two level fuzzy PRM for manipulation planning," *IEEE/RSJ International Conference on Intelligent Robotics and Systems*, pp. 1716–1722, 2000.
[16] G. Song, S. L. Miller, and N. M. Amato, "Customizing PRM roadmaps at query time," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2001, pp. 1500–1505.
[17] D. Hsu, T. Jiang, J. Reif, and Z. Sun, "Bridge test for sampling narrow passages with proabilistic roadmap planners," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2003, pp. 4420–4426.
[18] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2, 1999, pp. 1018–1023.
[19] A. Vargas Estrada, "Visualization tools for moving objects," Master's thesis, Texas A&M University, December 2005.