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

Aimée Vargas E. aimee@cs.tamu.edu
Jyh-Ming Lien neilien@cs.tamu.edu
Nancy M. Amato. amato@cs.tamu.edu

Technical Report TR05-011
Parasol Lab.
Department of Computer Science
Texas A&M University

September 21, 2005

Abstract

Comprehension of concepts and algorithms involved in the robotics field can be improved through the use of an interactive visualization tool. The motion planning problem consists of finding a valid path between a start and a goal configuration for a movable object or robot. The workspace, in traditional robotics and animation applications, is composed of one or more objects (called obstacles) and a robot. In this paper we present an interactive tool for visualizing and editing motion planning environments, problem instances, and their solutions to help students in their understanding of the motion planning problem in robotics related courses. 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 manipulation and evaluation of planner solutions and the animation of paths found by motion planners, it is specialized for randomized planners that use the Probabilistic Roadmap Methods (PRMs) or tree-based methods.

1 Introduction

Motion planning is the problem of finding feasible paths for movable objects among obstacles [1]. While planning motion is an every day task for us, the problem of motion planning is not as easy as it looks. As a consequence, there are many fundamental concepts in motion planning problems, such as degrees of freedom, configurations, and configuration space, that can be difficult concepts to conceive. In the classroom, a visualization tool that illustrates the motion planning problems, such as the shape of a robot, the environment that the robot is operating in, start and goal configurations and valid paths for the robot, can help students to more easily understand these concepts.

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.

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 factories and workspaces that they are not likely to explore without expensive equipment, such as inside an airplane engine or outer space. This can even include workspaces that are impossible to go to given the current technology, such as a prehistoric city.

Another good way to improve students' understanding of motion planning is to allow students to modify existing problems and play with the parameters of a motion planning strategy, and to allow them 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 of 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 can provide a means to operate on existing problems and create new problem instances. Teachers can take advantages of it because VIZMO++ allows users to change the properties of the objects in the scene such as 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 different functions offered in VIZMO++.

2 Related Work

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

2.1 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) [6]. 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 has embedded planners that can be used to attempt to solve a motion planning problem. Parameters for those algorithms can be configured through the interface. MSL 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.

2.2 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 represents the position and orientation 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) [7] 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 [8], 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, 9] or tree-based [10–12]. Notable examples of roadmap-based and tree-based planners are the Probabilistic Roadmap Methods (PRMs) [3, 4, 9, 13–17], and Ariadne's Clew algorithm [11] and the Rapidly-exploring Randomized Trees (RRTs) [12], respectively.

Probabilistic Roadmap Methods (PRMs) [9] are roadmap-based methods that generate a graph or roadmap where the nodes represent collision-free configurations and the edges represent feasible paths between those configurations. A roadmap may contain one or more connected component. The path will be a sequence of configurations that describe the set of movements that the movable object needs to perform to reach its goal. PRMs typically consist of two phases: roadmap construction, where a roadmap is built, and query, where the start and goal configurations are connected to the roadmap. A path can be extracted using graph search techniques.

3 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 L-tunnel environment which is composed of two L-shaped tunnels 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 visualizations, animation and manipulation. Since different planners generate sample distributions of configurations in C-space, roadmap visualization can help to better understand how each method works by looking at the distribution of the nodes and their connectivity of the roadmap.

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 latter, 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 use cases of VIZMO++ including:

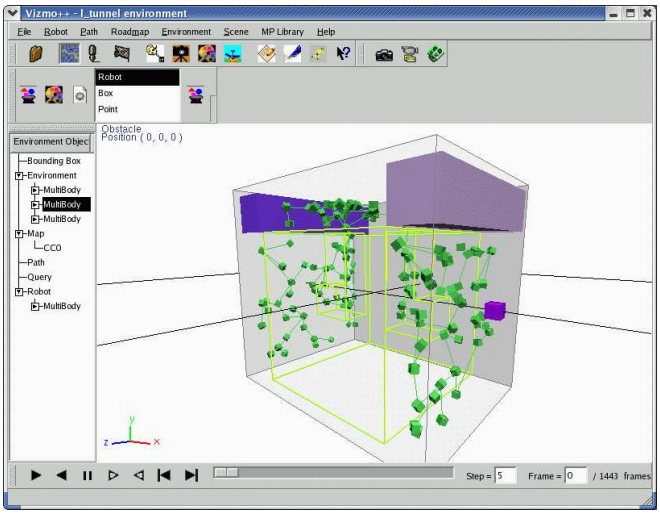


Figure 1: A snapshot of VIZMO++

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

4 Functionality of VIZMO++

4.1 *User interface*

The organization and distribution of the elements of the VIZMO++ 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 of VIZMO++ 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. Main menu bar also includes options for open and saving files, access to our motion planning library, auxiliary tools, and help.
- The toolbar placed at the top of VIZMO++ contains most used functions so users can access these functions 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 and therefore provides a quick access to these objects.
- An animation toolbar at the bottom of VIZMO++ gives access to the animation of the path.

4.2 *Visualizing the scene*

When VIZMO++ loads a workspace, students first can see the scene of the workspace, including a robot, one or multiple 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 modes, that can allow students to better understand and explore the environment. For example, some environments can be completely

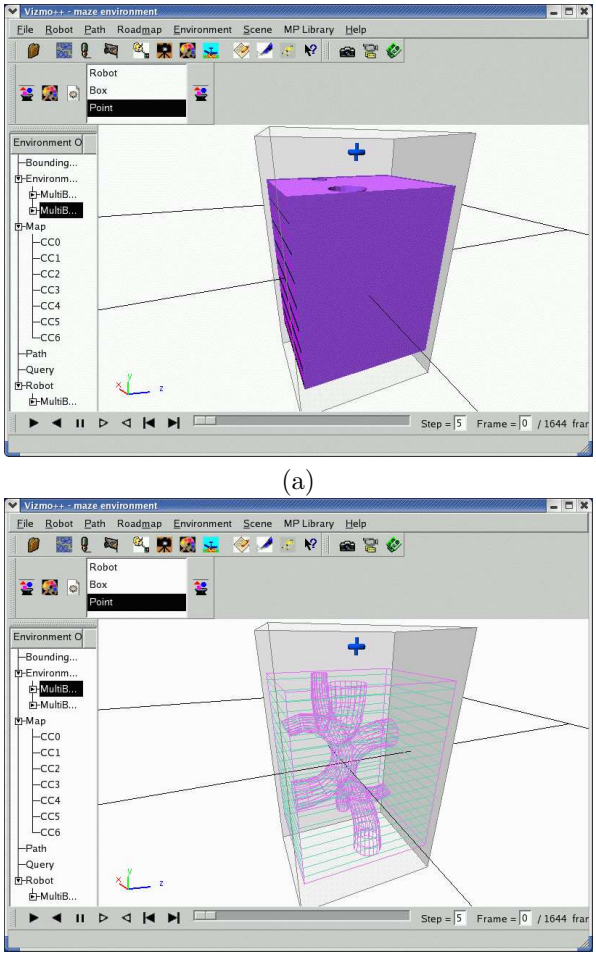


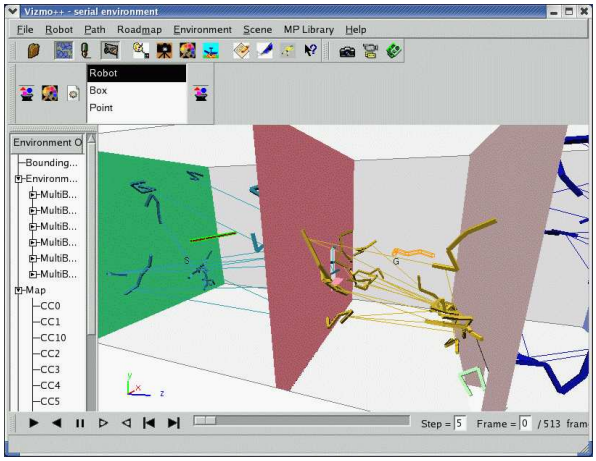
Figure 2: In (a) we show the solid model of the *maze* environment and (b) shows the same environment in wire mode

surrounded by walls and so the interesting characteristics of the environment can be hidden. In these cases, a wire mode can show the environment better. 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 in order to reach its goal, which is at the bottom of the maze. To change from one state to the other, the student only needs to select one or multiple obstacles and select a rendering mode.

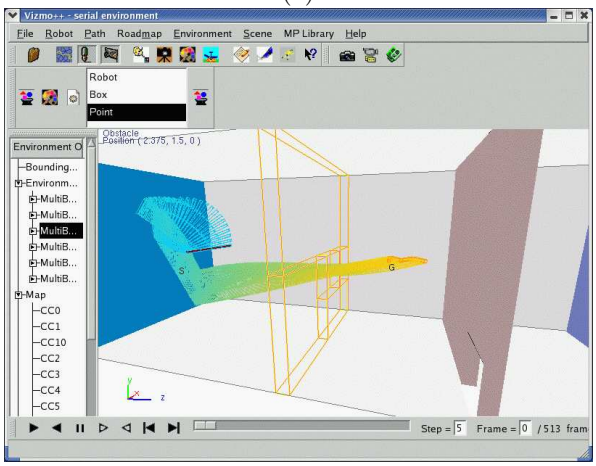
Other than these workspace objects, VIZMO++ also shows the query of a given motion planning problem. The query is shown in wire mode giving a different color for the start and goal configurations. Users can distinguish them not just by color but also for the labels: 'S' for start and 'G' for goal.

4.3 *Visualizing results*

The results generated by a motion planning strategy include a 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, boxes 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 is the slowest. The nodes can be scaled to have clearer visualization when the size of a roadmap is large. VIZMO++ renders the



(a)



(b)

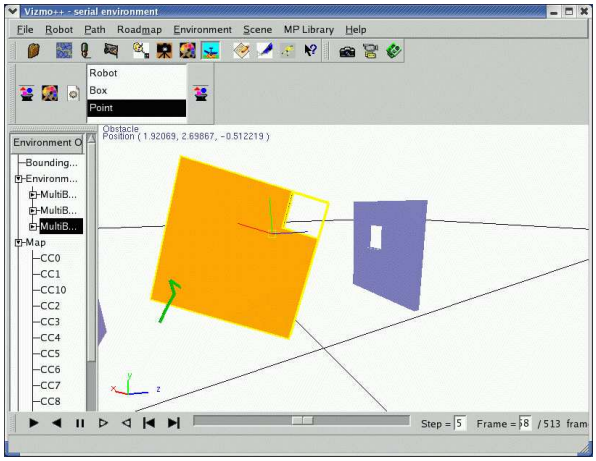
Figure 3: *walls* environment (a) shows the roadmap and the nodes rendered as actual robot configurations. (b) shows the swept volume of the path .

swept volume of a path and paths also can be animated.

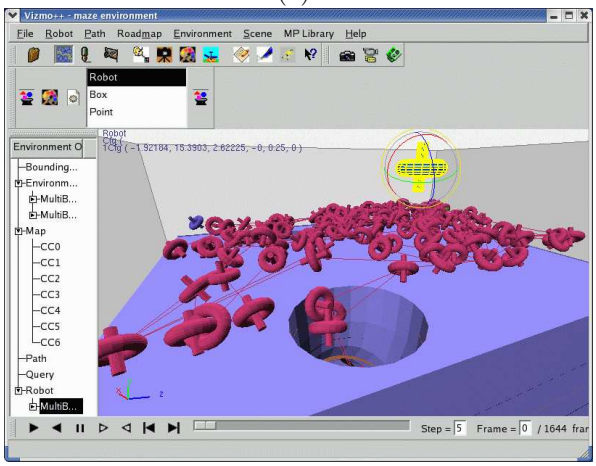
In Figure 3 we show the *walls* environment for a 9-DOF robot. The objective of the robot is to go from the first chamber to the second one and has to go through a narrow passage, which is a hole on the second wall. In (a) we show the roadmap with the nodes rendered as robot configurations. The connected 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 be displayed on the main window. In (b) we show the swept volume of the path. The path can be animated using the *animation* toolbar that is located at the bottom of the GUI. The student can play the path or can go frame by frame to see the robot's configuration in more detail.

4.4 *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 being moved around using the translation tool at the same time. The collision detection option can be turned on so users can know whether the obstacle is in collision or not 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



(a)



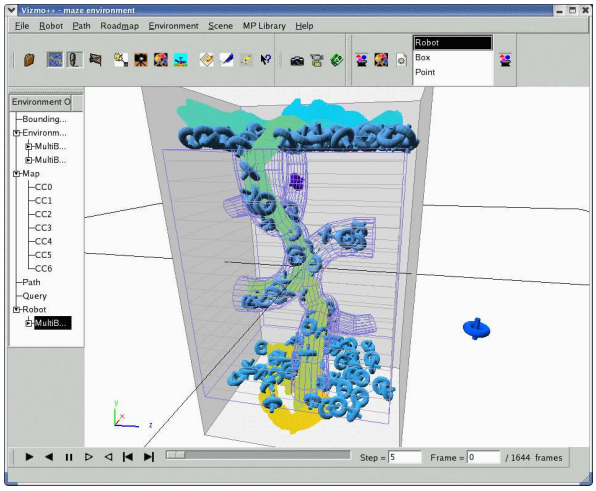
(b)

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

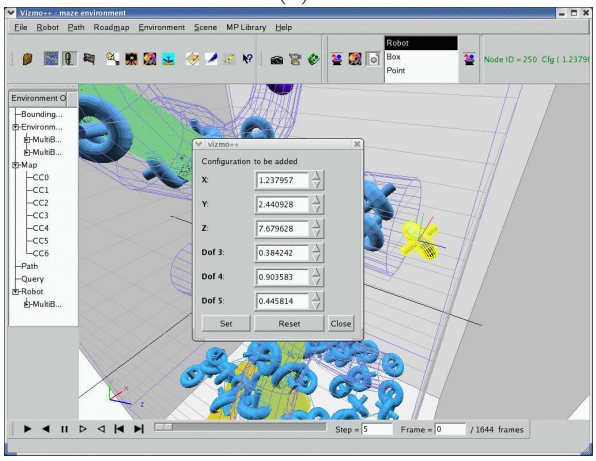
the student to create their own environments. Once an environment has been changed, the student can recompute the roadmap for the new environment by just clicking on the *automatic roadmap* button and then he/she can visualize the result.

4.5 *Editing roadmaps*

VIZMO++ provides tools to edit a given roadmap. A node of the roadmap can be selected to either see or modify its configuration. An environment can be composed of one or more narrow passages and this can prevent a planner from finding a solution. We enable users to add nodes to the roadmap. Nodes can be selected and copied, then using the GUI, users can change the new node's configuration and visualize how the configuration changes accordingly. Figure 5 (b) shows the GUI to add a node and change its DOFs. The node can also be selected and moved and the x, y, and z values will be updated in the GUI. Values for other degrees of freedom can be changed through a dialog. A user can generate a new roadmap by hand or can add nodes to an existing roadmap. VIZMO++ provides a function to check if the added nodes are in collision. 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.



(a)



(b)

Figure 5: (a) The swept volume of the path and the roadmap for the *maze* environment. (b) shows 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 VIZMO++ main window.

5 Case studies

5.1 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. First, the user needs to move aside obstacles to make space for the new one. To change the position and/or orientation of an obstacle, the user needs to first select it and and hit the *w* key to have access to the translation tool, or *e* for the rotation tool. Using the mouse, the user can now translate or rotate the obstacle.

2. Now that there is already an open space, the new obstacle can be added to the scene. The users needs to select 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.

3. The user might want to change the color of the obstacles and can do it randomly or 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.

4. Once the user is happy with the new environment, it can be saved to a file.

Creating a new query. In VIZMO++, the robot can be selected to be rotated or translated 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 rotation and translation tool described before. 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.

Figure 4 (b) shows the robot which is being rotated.

5.2 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 factories and workspaces that they are not likely to explore without expensive equipment, such as inside an airplane engine or outer space. This can even include workspaces that are impossible to go to given the current technology, such as a prehistoric city.

In this section, we show two environments in Figure 6. The first environment is an airplane engine model which contains very complex obstacles in workspace. The second environment contains a robot with a complex shape that represents a protein model.

5.3 Comparing different planning strategies

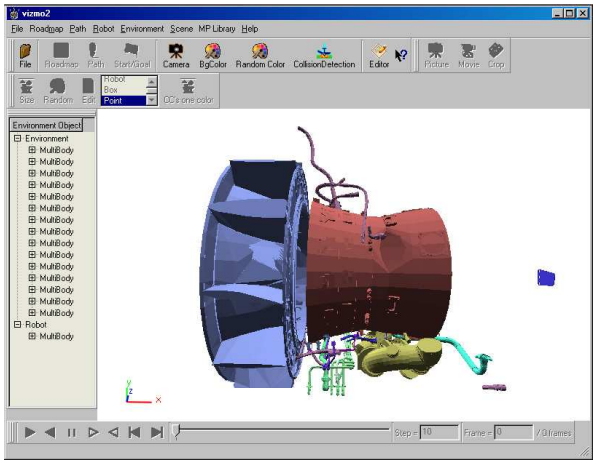
VIZMO++ can be used to help students better understand and compare different planning strategies. Figure 7 shows the *rigid* environment which is composed of two plates. The robot's initial configuration is in between the plates and its goal is to get out of them. We show in (a) the node distribution computed using the Bridge Test strategy [16] which generates samples between the plates and in (b) the roadmap computed using the Gaussian strategy [17] which generates samples around obstacles.

6 Conclusions

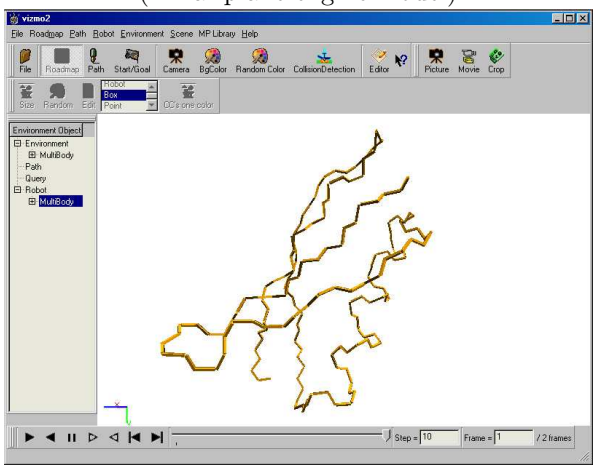
We presented VIZMO++, a tool for visualizing and editing motion planning environments, problem instances, and their solutions. VIZMO++ offers a self-explanatory graphical user interface that provides functionality that can help students in their understanding of robotics and motion planning concepts.

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 motion planning methods, easily 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 explain those concepts. We noticed students showed more interest when they were able to actually see what they were being taught.



(An airplane engine model)



(A protein G)

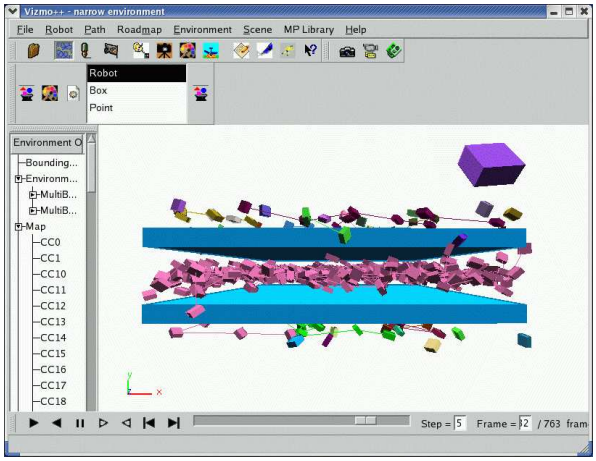
Figure 6: Examples of workspaces that are difficult to access.

7 Future Work

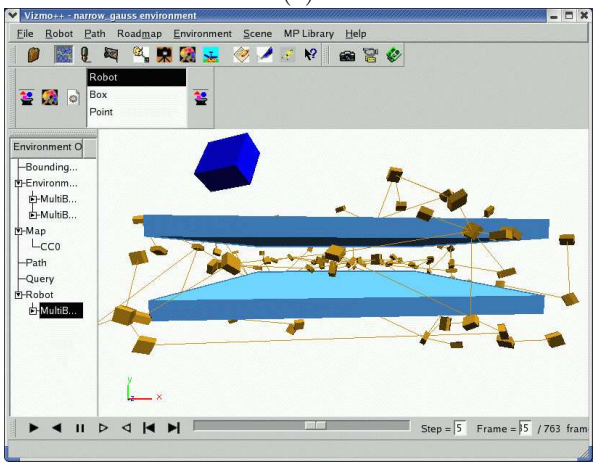
Even though VIZMO++ offers many functions that can be of help to students to better understand motion planning concepts, there are other functions we need to improve and work on. It can happen that a given problem may be dealing with more than one robot at the same time. Our current version of VIZMO++ supports just one robot so we will need to extend our current implantation 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 because for roadmaps with many nodes and connected components is difficult to distinguish what nodes and edges were used even though we offer the option of highlighting connected components. An *undo* function will be very useful. A student might accidentally delete an obstacle and currently, the only way of having it back is by opening the environment again.

References

[1] J.-C. Latombe, *Robot Motion Planning.* Boston, MA: Kluwer Academic Publishers, 1991.



(a)



(b)

Figure 7: For the *rigid* environment, (a) shows a roadmap generated with the Bridge Test strategy and (b) shows a roadmap generated with the Gaussian Sampling strategy.

[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] S. M. LaValle, "Motion Strategy Library," http://msl.cs.uiuc.edu/msl/.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 1999, pp. 473–479.

[13] R. Bohlin and L. E. Kavraki, "Path planning using Lazy PRM," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2000, pp. 521–528.

[14] 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.

[15] 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.

[16] 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.

[17] 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.