Walking Through a Maze

A Groovy Graphics Assignment

Edward Angel
University of New Mexico
Albuquerque, New Mexico
angel@cs.unm.edu

ABSTRACT

This assignment supports the key concepts in a first course in Computer Graphics for students in Computer Science and Engineering: modeling, geometry, transformations and interaction. The core of the assignment is to create a maze that can be walked through via the mouse or other interactive tools. The maze should have only one entrance and one exit. The walls are planes extruded from the floor.

CCS CONCEPTS

• **Computing methodologies → Fundamentals; Graphics and Interfaces; Modeling; Rendering**;

KEYWORDS

CGEMS, Computer Graphics Education, Groovy Graphics

ACM Reference Format:
Edward Angel. 2018. Walking Through a Maze: A Groovy Graphics Assignment. In *Proceedings of SIGGRAPH '18 Educator's Forum.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3215641.3215649

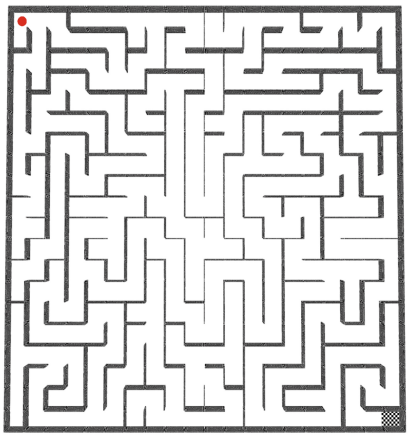

Figure 1: A Two-Dimensional Maze (Courtesy of Brad Key, UNM CS Student, 2014.

1 OVERVIEW

Virtually all first courses in Computer Graphics for students in Computer Science, Engineering and Mathematics have a programming component. By the middle of the semester students should be able to demonstrate their understanding of geometric concepts, modeling, viewing, transformations and interaction. A programming assignment at this point should not only require a demonstration of all these areas but should also encourage students to go beyond the basic requirements.* The core of the assignment is to create a maze that can be walked through via the mouse or other interactive tools. The maze should have only one entrance and one exit. The walls are planes extruded from the floor. Optionally there can be a ceiling. The construction of the maze can be done in 2D as a graph traversal problem and then the walls can be extruded to create the 2 1/2 dimensional maze. The minimum requirement for this assignment is to create the maze, display it in perspective from the entrance and provide an interface that allows the user to walk through the maze searching for the exit, continually updating the user's view. The application must prevent users from walking though walls. The construction of the maze can be done in 2D as a graph traversal problem and then the walls can be extruded to create the 2 1/2 dimensional maze. One algorithm (there are many) starts with an n x n array of square cells, each initially marked as unvisited. The algorithm proceeds by randomly removing walls between visited and unvisited cells. Upper class Computer Science students find this part of the assignment a nice application of their data structures and algorithms courses. Other students may need some additional background or some pseudo code. The maze construction can be presented as an earlier exercise in 2D as a way to introduce the API before discussing viewing. Figure 1 shows a typical maze with an entrance and exit chosen randomly. Note that with this type of maze, any cell is reachable from any other cell.

The maze is extruded to a 2 1/2 dimensional maze by converting each remaining line segment in the 2D to a rectangle. The challenge for viewing is use perspective to create an immersive feel as the user goes through the maze. Setting the viewing parameters with most APIs can be tricky. Moving the viewer through the maze can be done in many ways. The simplest is to allow use of the arrow keys to move forward or turn 90 degrees to the left or right. Better solutions use the mouse or a graphical interface and allow continuous motion. All interfaces must prevent the viewer from walking through a wall.

Rendering the maze can be as simple as just assigning colors to each rectangle. Most students prefer to either use texture maps or to apply lighting. Figure 2 shows a maze with texture mapping

SIGGRAPH '18 Educator's Forum, August 12-16, 2018, Vancouver, BC, Canada

ACM ISBN 978-1-4503-5884-2/18/08.
https://doi.org/10.1145/3215641.3215649

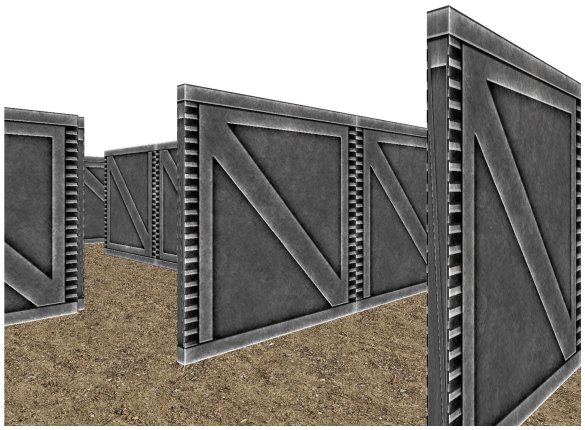

Figure 2: Three dimensional maze constructed from 2D maze by extruding sides and rendering each side with a texture. (Courtesy of Brad Key, UNM CS Student, 2014.

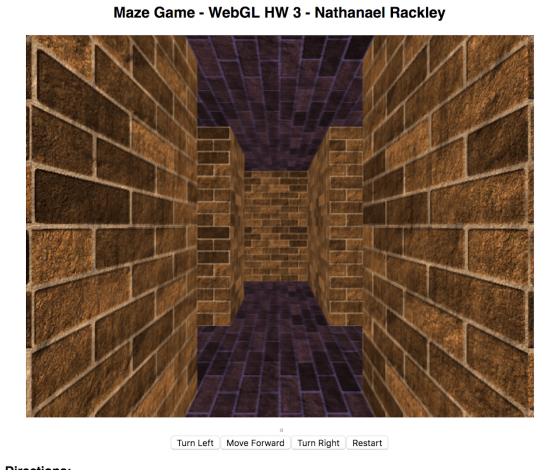

Figure 3: Maze with ceiling and floor (Courtesy of Nathan Rackley, UNM CS Student, 2014).

used for the walls and a floor. Figure 3 shows an example with a simple graphical interface.

2 METADATA

Metadata for the assignment is given in Table 1. The assignment can be used with students from computer science, engineering and mathematics who have a programming background equivalent to CS 2.

3 MATERIALS

The assignment can be done with any graphics API. Mostly recently I have been using WebGL. One of the advantages of WebGL (and three.js) is that the assignment can be done with any recent browser and students can employ a number of packages to construct interfaces. A side advantage is that students love to show their mazes to friends and family, something that can be done with via a laptop, a pad or a smart phone.

Table 1: Metadata in Tabular Form

Summary	Create a 3D maze with extruded walls with a single entrance and exit that can be âĂIJwalkedâĂİ through.
Learning Outcomes	Supports key computer graphics concepts including geometry, modeling, transformations and interaction
Classifications	Graphics, Interfaces, Modeling, Rendering
Audience	First course in Computer Graphics for undergraduates and graduate students in Computer Science, Engineering and Mathematics
Dependencies	Basic knowledge of OpenGL (WebGL preferred), possible to use three.js. Can be linked to a previous assignment in which the student creates a 2D maze using a tree traversal algorithm.
Prerequisites	First half of standard course in Computer Graphics that includes a programming component
Strengths	The assignment reinforces all fundamental concepts, allows students to add many options and is fun. It can be done with any API.
Weaknesses	Maze generation may be difficult for non CS students
Variants	Add texture/lighting, extend to a 3D maze
Assessment	Can be assessed by other students if done using WebGL or three.js. Basic requirements are easy to test but assignment allows students to show creativity to go beyond the basics

ACKNOWLEDGMENTS

I am indebted to three of my students from 2014, Brad Key, Nathan Rackley and Stephen Harding for allowing me to show their work.