Assignment 2

CSE251 - Graphics, Spring 2017

Due: Feb 11th, 5 PM.

1 The Problem

The previous assignment dealt with a 2D world. Now, we shall move to 3D. In this world, there is a movable cuboidal block. The goal of the player is to make it fall into a square hole. Facility should be provided to move the block, both, by mouse and keyboard. The game can also be seen from multiple views and facility must be provided for these camera views to be controlled both by mouse and keyboard.

The requirements mentioned are minimal, you can be creative and add interesting features.

2 The 3D World

The world consists of square tiles which can be arranged in any manner. The tiles have to be carved out of a 10 X 10 area, having a minimum of 50 tiles. As evident from the image given, some of the blocks in the 10 x 10 area will not have tiles, therefore making the game more difficult to play. The player should aim to put the block into the final square hole with a minimum number of moves.

The movable block is a cuboid made of 2 cubes stacked on top of each other. The length of the side of the cube is same as that of a tile. The Player wins the game when the block falls through the square hole (the block has to be vertical for it to fall through the hole). If the block falls down the edges then the game ends.

The tiles are of 4 types - normal tiles, fragile tiles, bridges and switches. In order for a bridge to be constructed, a switch (in a different location) needs to be pressed by the movable block. Bridges can have 2 states - open and closed, which can be toggled as per the requirements of the player. It is not necessary to remain on the switch to keep the bridge in its current state. The next kind of tiles - fragile tiles, are marked using different colours. If the block stands up vertically, then the tile breaks. To move on fragile tiles, the block should lie horizontally (on the longer side).

Figure 1: Mock-up of the sample world

The number of moves made and the time taken should be displayed in the game view.

3 Controls and Camera

The movement of the block should be controlled by both, mouse and keyboard. The block can only skip one row/column (one cell, in any direction) in case of jump. For uniformity, use arrow keys for the movement of the block in a particular direction. Define other keys as per your convenience and mention them in a Readme file. Additionally, the following camera views have to be incorporated:

- *Block view:* This is a view from the block's position where only a part of the world in

front is visible. In other words, in this view, we see what the block sees, as if we were the block.

- *Top View:* This is a top-down view, as if we were looking vertically downwards from a position in the sky. This gives a clear picture of the path.
- *Tower view:* Here, the camera is sitting on a tower, to the side of the plane of playing, observing it at an angle.
- *Follow-cam view:* This is a view of the block and the region in front of it from a location behind and above it, as if the camera is following the block.
- *Helicopter-cam view:* Here, the camera is movable with the mouse in an intuitive manner. Clicking and dragging should change the look angle, the up vector should remain up always, and the scroll wheel will move the camera closer or farther away from the scene.

4 Optional

One can also include different difficulty levels or stages. You can design complex and interesting paths (for eg: only one way to reach the hole). Further, you can experiment with more camera views, audio, animations for movement/winning/losing, textures, etc. You may also add any additional items that you feel would make the game more interesting. Be Creative. List out the additional features that you have included, their behavior, and how to control them in your game, in the Readme.

5 Submission

You submissions should include your source code, a makefile and a compiled executable. You need to include a readme file that describes any additional information that is needed in compiling/executing you code. Do not use any non-standard libraries. In addition to these, include a file named help.txt or help.pdf (no word or other proprietary formats) in the submission that gives a one page description of the game and how to play it. Please make sure that your codes compile on the server before submitting, to ensure compatibility and uniformity.

6 Grading

You will be graded based on the correctness and efficiency (speed) of the implementation of the minimum elements described above. This will contribute to 90% of your grade. Remaining 10% will be given based on the improvements that you do over the basic game. In addition, submissions that are found to be exceptional by the graders, will be showcased, and will be awarded extra credits up to 10%.

7 References

This game has been inspired from Miniclip's Bloxorz. You can refer to it, for the basic outlay of the world, controls, and more interesting ideas to implement.