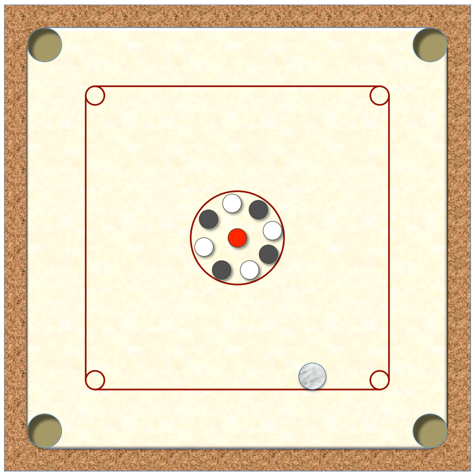
CSE251: Graphics - Spring 2016:

Assignment 3: WebGL Board Game

Due Date. March 28th, 10 pm.

1 The Problem



The goal is to create a board game that can be deployed through your website and played on your browser. The suggested game is carrom.

In the following sections, the minimum requirements are mentioned. You may enhance the game as per your liking.

You should provide a single page quick start guide to those who want to play the game (aka TAs), describing the additional controls (basic controls should be as described below), and additional features.

2 The Game

The board consists of a flat square of size $75cm \times 75cm$ with holes of diameter $5cm$ at the four corners. You have one red, 4 white and 4 black circular coins of $3.5cm$ dia. and a striker of $4cm$ dia. All coins are $1cm$ thick.

The initial position of the coins are as given in the figure. Your goal is to use the striker from your side to pocket all coins of one colour without pocketing coins of the other colour. Your score starts at 100 and reduces by 1 point every 5 seconds and by 20 points for pocketing a coin of the opposite colour or the striker. You get 5 points for pocketing a coin of your colour. If you pocket the red, you get 20 points if you follow it with a coin of your colour in the next attempt. If not, the red goes back at the centre.

You should be able to control the location of the striker and the direction and power of the strike. To play, you place the striker at any point on your base-line. You should be able to control this by a click of the mouse and/or using the left-right arrow keys. Hit the enter button to finalise the striker position. You may also allow click and drag to reposition the striker.

Once the striker position is finalised, you select the direction of strike. This is again done by the mouse or left/right arrow keys. The direction selected should be indicated to the user using a straight line from the striker. Once again, use the enter key to finalise the direction. Finally, select the power of strike using the up/down arrows or using the mouse wheel. Indicate the power level to the player using a scale on the right side. Once the power is selected, initiate the strike using either the enter key or a mouse click. Alternately, you may use the mouse cursor position to determine the direction and its distance from the striker to determine the power and a click initiates the strike.

The motion of the coins are governed by laws of motion. Assume perfect collisions with the sides of the board and coins and use laws of reflection to compute motion directions and velocities. Also assume a constant friction while moving and that the coin stops if the velocity is below a threshold. Other controls may be defined as per your imagination and described in your readme file.

3 Camera

One should also be able to control the position of the camera to the following positions:

1. Top View: A simple view from directly above, looking down. You should be able to see the whole board in this view.
2. Player Cam: Here the camera is positioned on the player's side pointed at an angle towards the centre of the board. The whole board should be visible.
3. Coin View: View from the coin being aimed at. You should be able to see the striker strike the coin and the camera should move along with the coin if it moves.

Note that one should be able to switch views, and then control the cameras and the person as per their wish.

4 Optional

Feel free to include additional objects, animations, textures, etc. to make the world more realistic and rich. You may also include a replay if a coin is pocketed. Additional interesting camera views may be provided. You may also add sound effects, which can significantly improve the user experience.

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.

The final submission, on Mar. 28th, should contain the complete code with the corresponding readme file.

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 world. In addition, submissions that are found to be exceptional by the graders, will be showcased, and will be awarded extra credits up to 10%.