2D Transforms and Rendering

This assignment will be 10 of the 55 points available for the programming assignments.

Notes for CSE 470/598
Instructor: Ross Maciejewski

1 Project Guidelines

- Have fun, learn, be creative, and create something you are proud to show-off!
- No teamwork allowed!
- Comment your code - you will be glad you did. You will be reusing code throughout the semester.
- You will upload your projects on myASU.
- If you have **good reasons** that you cannot complete a project on time **and** you have written documentation, then we can make adjustments to due dates. However, you must notify us before the due date that you would like to discuss such an arrangement. Good reasons would be illness, family emergency, visiting a conference to present a paper, …

2 Upload your projects

- Check the due date of the project on myASU
- Use myASU (digital dropbox) to upload your project. (Do not send the program as email attachment to the instructor or TA!)
- Your project should be packed in a zip file and named with the following naming convention:
  PX.Lastname.V1.zip
  (if you submit some updated files, use a different version)
  So my second project would be named
  P2.Maciejewski.V1.zip
  If I then found a last minute mistake and want to upload again I would use the name P2.Maciejewski.V2.zip
- If there is someone with the same last name in the course you can add your first name after the last name in the filename.
- The zip file should include all the source code and should be ready to compile. That means it should include your Visual C++ project files.
- The zip file should include an executable and the glut32.dll so that it can be started directly by clicking on it. (from wherever the file is downloaded to).
- A short text file called "instructions.txt" – this text file should include the following:
  - a) instructions about how to navigate in your program, including the function of special keys (e.g. x exits the program)
  - b) special functions that you implemented
  - c) changes from the original specification that you negotiated with the instructor or TA to the original specification
- You can use rar instead of zip to compress your files

3 Project 2

3.1 Purpose

This project has been designed for you to do some fun 2D rendering! You will be using translations, rotations and scale.

3.2 Overview

You should take the sample code provided on blackboard and compile the code. You are building from your Project 1.

3.3 Required Functionality

Here are the elements that your program must have. Despite this "laundry list" of requirements, there is some room for creativity! Note: If the text suggests a menu item "Sound - Off" that would mean that "Off" is an entry in the submenu "Sound".

- **(0p) Take two new images** – You will take two new images. One is an index card of a solid color with a black border on a surface similar to 1.png (no rotation of the card). The other is of yourself holding an index card of a solid color with a black border. The index card should be held at an angle, but also approximately flat in the picture. This will be similar to 2.png in the homework folder. However, your face should be in the picture. A camera phone will work perfectly fine, ask a friend for help taking the pictures. You want nice lighting conditions. All images will now be oriented correctly. Code will be provided on blackboard for that function.

- **(5p) Load the two new images** – You will create a menu entry "Images" with subentries "1" and "2". When I click on "1" the first new image you took will be loaded on the screen. If I click "2" the second new image will be loaded. For simplicity, the images should be the same size and orientation. (Note that I can go back and forth between these as often as I would like and if you don't do the 0 point part above you'll get 0 for this part). Menu options from Assignment 1 should be removed (except for exit).

- **(20p) Finish the renderStickMan function** – You will create a hierarchical object "stickman" consisting of a head, arms (1 line each) and legs (2 lines each indicating a knee joint).

- **(10p) Draw many stickmen** – You will enable the program to create up to 3 stickman. Each time a user clicks, a new stickman is drawn such that his waist is at the point that was clicked. The stickman will then begin moving. If a user clicks more than 3 times, the oldest stickman disappears and a new one is added. You must use an appropriate data structure for this.

- **(25p) Animate stickmen** – You will need to create variables for your stickmen to animate them as shown in the figure below. Your stickmen should perform each a cycle of the action once every two seconds. You will need to set up a timer to accomplish this. For the undergraduate portion you may assume your stickman is always moving on a white background and has a white bounding box behind him.

- **(15p) Graduate Students –** You will create a menu option "Filled" with submenus "on" and "off". If Filled is "on", then you will automatically detect and fill in the index card in your image with a blue color. For this, you will define a "best fit" rectangle and draw a scaled and rotated rectangle over the index card. If turned off, the image should return to its normal appearance.

- **(0p) Graduate Students –** You will create a menu option "Stickman" with suboptions "Default" and "walking". In the default mode, mouse clicks will work as described for the undergraduate portion. If you don't complete these menus you will lose 10p.

- **(25p) Graduate Students –** In the "walking" mode, you will erase all other stickmen and create a stick man that walks along the lower edge of the index card. His joints should move somewhat realistically and if on an incline should walk up the hill. When he reaches the end of the index card, he will turn around and walk back. Thus pacing across the card. Again, a timer should be used to insure proper animation rates. Style points will also be counted for the walking here.

Total points are out of 100. Undergraduate points scale out of 60 (so 60/60 = 100/100 for undergraduates). Undergraduates may earn an additional 10% maximum for completing the graduate portion (thus earning 110/100).

3.4 Grading

These points represent an initial weighting of the functionality. **Important:** You get full points for a *good* implementation of the required functionality. In principle you will get full points if you implement the requirements correctly, but there are many ways to do things according to specification that we might consider *unreasonable* or *undesired*. Especially we will deduct points if the design choices make it difficult to judge the program (E.g. you choose materials that are so dark that it hard to see anything, the objects move too slowly or too fast, the control of the program is extremely difficult.)

Note: we might change the weights slightly if a requirement is misunderstood by a majority of the class or some other special circumstance. I do not expect that to happen though.