

Computer Graphics 50:198:456/56:198:556 (Spring 2009)

Homework: 2	Professor: Suneeta Ramaswami
Due Date: 3/4/09	E-mail: rsuneeta@camden.rutgers.edu
Office: 321 BSB	URL: http://crab.rutgers.edu/~rsuneeta
	Phone: (856)-225-6439

Programming Assignment #2

The goal of this programming assignment is to write a moderately complex interactive 2D graphics program. In particular, you will modify the paint program (`newpaint.c`) from Chapter 3 of the text. You will need to use menus, mouse input processing, and a simple data structure.

Problem Description: The key difference between the paint program given in the text book and the one that you are asked to implement in this assignment is that the user will be able to manipulate and edit the objects drawn in the window. In particular, the user will be able to delete, move, and reshape the objects. This means that it is now necessary to save the objects drawn on the screen in an appropriate data structure. Further details appear below:

- The paint program allows the user to interactively draw points, line segments, triangles, and rectangles (you may eliminate the text writing option of the original program).
- The layout is a 500×500 window. As in the original program, rectangular icons for the four figure types should be displayed on the upper left corner of the window.
- You may ignore the issue of resizing the window.
- The mouse button functions are as follows:

Left mouse button: Used for selecting the figure to be drawn (as in the original paint program `newpaint.c`). This is done by clicking on one of the icons in the window. The user then draws the object by clicking (with the left mouse button) the correct number of vertices.

Middle mouse button: Used for activating a menu. The menu should have six items:

1. *Colors*, which is a submenu with eight possible colors in which a figure may be drawn (as in original program).
2. *Pixel size*, which is a submenu to increase/decrease the “size” of the points drawn (as in original program).
3. *Fill*, which is a submenu with ‘fill on’ and ‘fill off’ options (as in original program).
4. *Edit*, which is a submenu used to modify existing figures. It has the following options: *Delete*, *Move*, and *Reshape*.
5. *Clear*, which clears the window (i.e, deletes all objects drawn in the window).
6. *Exit*, which causes the program to terminate.

Right mouse button: Used to modify (delete, move, or reshape) an existing figure. Note that this mouse button will be used after an appropriate selection has been made using option 4 of the menu attached to the middle button. If such a selection has not been made, there should be no action when the right button is clicked.

- *Delete*. The user clicks on the figure to be deleted with the right mouse button. If two or more figures overlap at the position where the mouse is clicked, the figure on top (the visible one) should be deleted. If there is no figure where the mouse is clicked, there is no action.
- *Move*. The user moves a figure by clicking on it with the right mouse button and holding it down while moving the mouse to the new location. As above, if two or more figures overlap at the position where the mouse is clicked, the figure on top is moved. If there is no figure where the mouse is clicked, there is no action. The user should see the object move as the middle mouse is held down and moved (use the callback `glutMotionFunc` appropriately). Note that when a figure is moved, it is considered to be the most recently drawn figure (i.e., it will obscure any figures on which it is placed).
- *Reshape*. The user reshapes a figure by clicking on a vertex with the right mouse button and holding it down while moving it to the new position. The user should see the object’s changing shape as the middle mouse is held down and moved (as above, this can be done with a `glutMotionFunc` callback). As with the move option, reshaping a figure should make it the most recently drawn figure. See Figure 1 for examples of how objects may be reshaped. Note that the reshape option does not make any sense for points, and for our purposes reshaping a point is the same as moving it. A note of caution about choosing a vertex for the reshape function: It may be difficult to pick out the exact coordinates of a vertex by a mouse click. The better option may be to allow the mouse click to be “close” to a vertex, such as in a 3x3 box centered at the vertex.

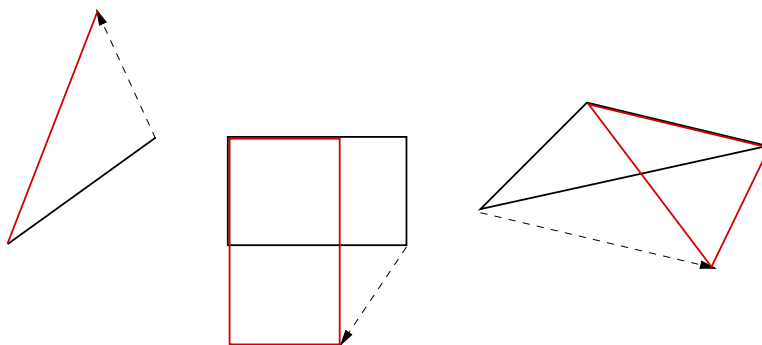


Figure 1: Reshaping figures: The arrow indicates the new position of a vertex. The original figure is drawn in black, and the reshaped one in red.

The *edit* option should stay active until one of the other mouse buttons is clicked. For example, if the user chooses the *delete* figure option, successive right mouse button clicks should cause successive deletions. However, as soon as one of the other mouse buttons is clicked, the delete operation (or other edit option) should become inactive.

To provide the above functionality in your program (which allows the user to pick objects from the graphics window), you will need to store everything that has been drawn interactively by using a suitable data structure (lists should suffice). My suggestion is to create a new class for figure objects. All drawn objects can then be stored in a list of figures, which can be manipulated, for

instance, with the `list` type of the Standard Template Library. However, implementation details are left to you and you will not be graded on the specific choice of data structure.

Required for graduate students, extra credit for undergraduates: In addition to a line segment, rectangle, triangle, and point, allow the user to also interactively draw a circle. This can be done by first clicking the center and then another point to determine the radius. Note that a circle can be approximated as a regular polygon with a large number of vertices. All the edit functions described above should work for the new object as well.

Grading: The assignment is worth 100 points. For undergraduates, the extra credit part is worth 20 points. A program that implements the required functionality and moreover, is well-designed will receive full credit. Needless to say, most of the new functionality in this modified paint program comes from the *Edit* functions, and hence this will have the most weight for your grade. Haphazard program design will cause you to lose points even if your program implements all the features.

Sample executable: The executable of a sample program (a past homework submission by a student) is available in the shared class directory `/class/rsuneeta/cgs09/`. The program, called `hw2`, was compiled on the linux partition of my laptop. Copy it to your home directory and run it. Note that the program uses only the left and right mouse buttons because the student wanted to design the program for a two-button mouse. You may do the same if you wish.

What to turn in: Email me the **final version** of your program before midnight on March 4, 2009.