Program is submitted on time (late submission subject to as much as 100% deduction)

Working Makefile included

Program builds without errors or warnings

Program is readable, modular, well documented, and reasonably efficient, as well as correct:

* Readable (20)
  + Meaningful variable names 5/5
  + Meaningful function names 5/5
  + Limited use of obfuscated code 5/5
  + Appropriate internal comments 5/5
* Modular (10)
  + Functions perform a single task 4/5
  + Classes are general and reusable where possible 5/5
* Well documented (12)
  + Doxygen main page describing overall program 3/3
  + Classes described 3/3
  + Class member functions documented 3/3
  + Doxygen runs without errors or warnings 3/3
* Reasonably Efficient (10)
  + Program uses appropriate data structures and algorithms 5/5
  + Program does not leak memory 5/5
* Correct (88)
  + Converts input image from 24 or 32 bit to greyscale 2/2
  + Converts greyscale image to binary threshold image 2/2
  + Display shows greyscale image 2/2
  + Display shows binary threshold image 2/2
  + Display shows connected components image 2/2
  + Display shows filtered components image 2/2
  + Right / left arrow keys decrease/increase binary threshold value 4/4
  + Up / down arrow keys increase/decrease filter size 4/4
  + Displays refresh when right, left, up, down keys are pressed 5/5
  + Image labels are toggled on/off using space bar 5/5
  + Four images are correct for SideHead.png file (128 thresh, 32 filter) 16/16
  + Four images are correct for headtop.png file (64 thresh, 64 filter) 16/16
  + Four images are correct for text.png file (100 thresh, 25 filter) 16/16
  + Four images are correct for chest.png file (225 thresh, 25 filter) 10/10
* Bonus : Turned in on Friday(8) 8/8
* Checkpoint #1 (seg fault) -3
* Checkpoint #2 (displaying incorrect image) -3
* Removal of globals from example code +5

Code Review – A place for comments from just reading the code and noting good, bad, ugly, and interesting features. Can account for as much as 10% of the total grade 10/10

Total Points: 148/150

* When you have a large switch statement like the one in your KeyEvent::action member function, it indicates you should be thinking about better polymorphism. Deriving some additional key events would clean this up considerably.
* Yikes! The formatting in initOpenGL is fairly ugly, did some tool do this (astyle?). Also, the messing around with the title string for the window creation call is unnecessary. Take a look at stringstreams, you could have done something like this:

stringstream ss;

ss << “Threshold: “ << Threshold << “ Filter Size: “ << filterSize;

glutCreateWindow(ss.str().c\_str());

* Good job creating a separate disjoint set class.
* Extra super good job removing those nasty globals.
* Your program exactly matches the reference in terms of the four test files. Excellent work.
* Good job of making the functions (mostly) short, responsible for just one thing, easy to read. Callbacks nice and clean as well. The one exception is the AnalysisEvent functions. A couple of those got a bit out of hand
* Good descriptive comments in your documentation especially in the AnalysisEvent class
* There is some memory leakage but it appears to be attributable to lodepng and openGL/glut
* All around great job you two. I see that you used git branches to manage development, good to see that.
* I could nitpick the code a bit here and there but none of us have time for that. Overall it’s a good job. Well done!