Course homework structure

This assignment is worth 20% of your course grade. It is due on 10/21, the original date of when the proposal was due. The proposal, worth 10% of your course grade, is delayed a week to 10/28. A second assignment, to be due November 16, will also be worth 20% of your course grade. The final paper of 20 pages, or the final project with 10 page write-up, is worth 50% of your course grade and is still due 12/14. There still are no exams.

New course imetable and summary:

- Thurs, October 7: Assignment 1 available
- Thurs, October 21: Assignment 1 due = 20% of grade
- Thurs, October 28: Proposal due = 10% of grade
- Thurs, October 28: Assignment 2 available
- Tues, November 9: Teaming agreements fixed
- Tues, November 16: Assignment 2 due = 20% of grade
- Tues, December 14: Paper or project due = 50% of grade

Visual Combination Lock: Design Specifications

The goal of this assignment is to take a short sequence of visual images, and to determine from them if the user has placed some body part(s) in a predetermined sequence of locations. For example, the program can ask for two images of the user’s hand on the table, and decide if the hand is in the upper left in the first image, then in the lower right in the second. Many other variants are possible, and, in fact, part of the grade will depend on how creative the domain engineering and the grammar has been.

To do this assignment, you will need an account in the CLIC lab, which is CS 486, accessible round-the-clock from the Biology entrance on main campus. You need to do your programming in C or C++ on one of the dozen or so SUN workstations there that have a camera. You will also need some software that is provided by Sun, or by other sources for the course. You should check the web page for more details on the code that is available.

To help structure the assignment, it is broken down into four steps with equal credit, with the full assignment worth the 20 points toward your final grade.

1 (For 5 points): Domain engineering step.

First, to get a feel for what the images and the domain looks like, use the SunVideo program which is on the web page. This displays in a window on the workstation what the camera sees. The images are 320x240 full color, real time. But that’s much more than you need.

So, using the code available from Sun and indicated on the web page, capture only several individual images of a body part—it should still be attached to the body it belongs to, of course—against the background of your choice. The binary file of the program for doing so is on the web page. (Source code is also available for those more adventurous in one of the common directories; ask the instructor for more details.) The program captures a single frame as a 320x240 pixel color JPEG format file, which means it is unfortunately compressed. The command line is:
\texttt{rtvc_capture_movie -f 1 -C Jpeg -o [filename.jpg]}

Note that the "f" flag specifies the number of frames; you want what is essentially a very short movie of a single frame. The "C" flag can be used to get other formats, but JPEG is fine. By using a shell script with the "echo" and "sleep" commands, you make a driver for this program that alert the users that you are about to take N successive single frames, and indicate each one.

To view these files, you can use the program called "xv" (full path is: /usr/local/X11/bin/xv)Click within its window using the right mouse button to get a menu, and type "i" within the window displaying an image to get information about the image.

But to manipulate these images, it is much better to convert them to PPM format, which uncompresses them and separates them into red, green, and blue images. One tight file then becomes three larger files, but they have a simpler format. You can use "xv" to view these, also. To do the conversion, use "man ppm" and/or "man pgm" to give you information about the program "convert" (full path is: /usr/local/X11/bin/convert) which does the job:

\texttt{convert [filename.jpg] [filename.ppm]}

2 (For 5 points): Data reduction step.

Next, use the code on the web site, which was written for the Computer Vision course (CS W4731), to define the structure of the PPM files (this is in the ".h" header files) and to access components of the files (this is in the ".c" method files). This works in either C or C++, and you are free to choose either. Find a way to manipulate the images to get a good binary image of the body part and then to determine the (x,y) coordinates of its center of mass. If you wish, you can clean up the image in various ways, but good domain engineering should make much of that unnecessary.

3 (For 5 points): Parsing and performance step.

Define the grammar for handling the symbolic data derived from the imagery. This would require a definition of tolerances (what, exactly, does "upper left" mean), and a clear documentation of why these decisions were made. If the grammar is more complex (for example, it can include symbols that indicate "reset", or various "counts"), it must be documented.

You must run the grammar on at least 10 different sequences. At least seven of these sequences should run correctly (that is, the system must give the proper answer), and at least three of these sequences must indicate a system failure of some sort. (Truly, this will not be hard to generate.) For the failures, you must explain what failed and why--and how you determined what caused the failure.

To document this step, you should use "xv" to create printable--and probably reduced--images of your input. Thus, what you turn in must be not only your code and your system's decisions (e.g. "yes, that is the combination", or "no, that is not", or, "oops, I am confused"), but also some record of the images used in that particular sequence.

4 (For 5 points): Creativity step.

The default definition of this problem is the one given above: two images, one with a hand in the upper left and one with the hand in the lower right. Doing this perfectly with the first three steps gets 15 points. However, to get full credit for the assignment, you have to do something else in addition. For example,
you can use the user's head, or head and hand in combination; you can allow the domain to vary in some way; the combination lock could include a reset signal, or some decoy positions, or ways to signal which part of the sequence "really" should be processed; you can use relative positions rather than absolute ones; you can use poses that vary in area (e.g. closed fists, karate chops, flat palms), etc. Whatever variation is chosen should affect the grammar for parsing, and it should be documented in the code. A warning: it is still necessary for the system to work at least seven times, so don't try to be too creative.

And, as a general rule, whatever you do in code or in writeup, style counts. It is your obligation to write it all up so that the instructor and/or the TAs can understand it on the very first try.