project AVRiL

Monthly Progress Report

December 2006

Advisors:
Dr. Sohaib Khan
Dr. Umar Saif

BSc Sproj '07 - Group 7:
Ahmad Humayun (CE / 2007-02-0236)
Ozair Muazzam (CS / 2007-02-0162)
Tayyab Javed (CS / 2007-02-0130)
Yahya Cheema (CS / 2007-02-0370)

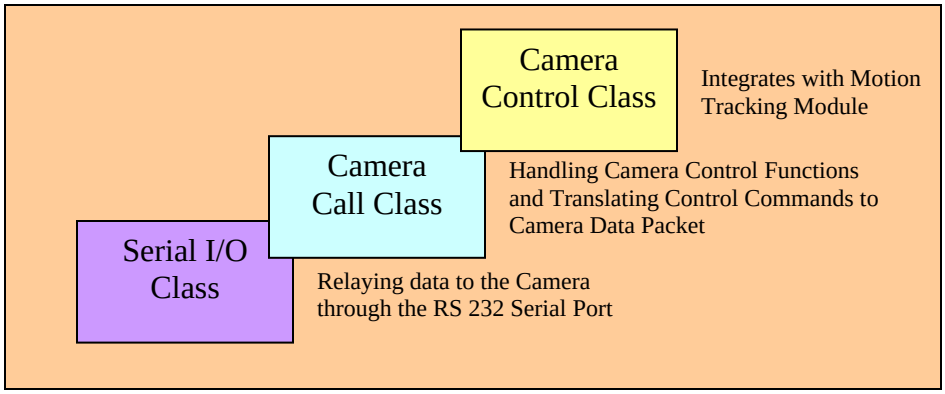
avril.sproj.com

1. Components completed

a. Camera Controller

One crucial part of the project was interfacing the camera with the computer that we would be using. The camera used was an ELMO 150S PTZ (point-tilt-zoom) camera, which was connected to the computer through RS 232 serial port. The video output from the camera was received through a PCI digitizer card, installed on the same PC giving the commands.

Camera Control Module



Using the datasheet provided, we looked at all the raw functions that could be called on the camera, and used the C++ library provided with the camera to call and test the raw functions of the camera in order to pan, tilt and zoom the camera.

Since these functions were direct calls to the camera, they were of a very hardware-oriented nature, and multiple calls had to be made to achieve a simple goal. To simplify the process of using these functions of the camera throughout the course of the project, we built another class on top of the library provided. On this class, we could call intuitive functions like moveUp(int amount), moveLeft(int amount). This class would be the interface to the Motion Tracking Module.

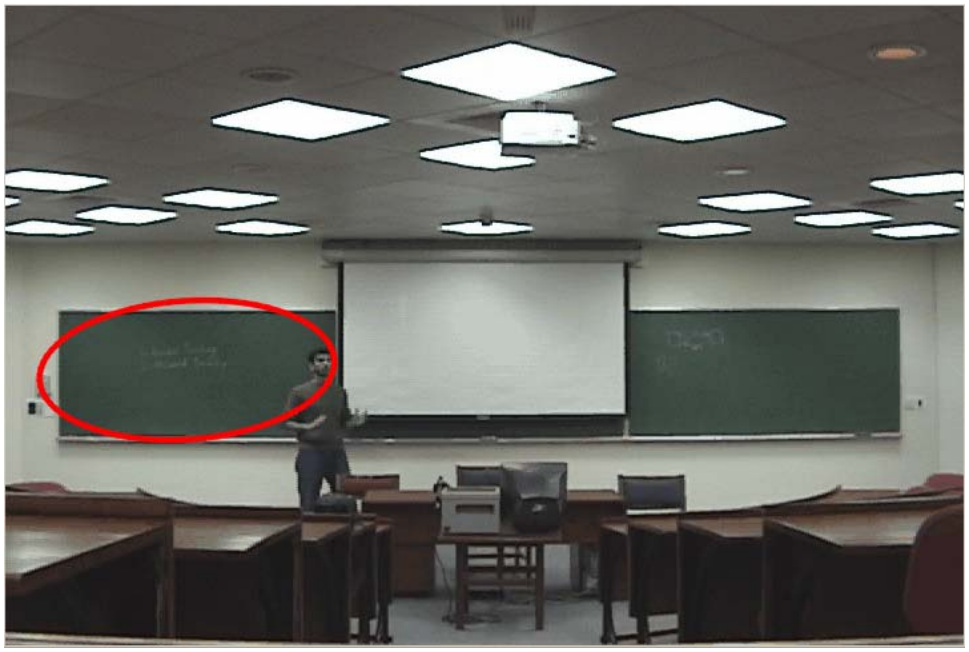
We then tested the class to see if it will be able to handle the requirements of our project, and it had simplified and satisfied most of the functionality of the camera as far as the requirements of the project were concerned.

b. Motion Tracking

Motion tracking is one of the most important components of the project. Therefore, we decided to get started with it in the first phase of the project. To keep things simple, initially we used basic general motion tracking algorithms without any heuristics to take advantage of features intrinsic to our problem such as a constant background (background subtraction) and features of the lecturer (face detection, human form detection) that can be used to simplify tracking.

We ran the CamShift algorithm and found that after initially marking the lecturer, the algorithm failed to follow him properly.

This video we used was recorded in an auditorium within LUMS, with the camera held static, facing the board as the wide angle camera would do in the later stages of the project. We also had a person act as a lecturer and move around as an instructor would.



-Tracking without background subtraction

After initially marking the instructor at the left-end of the frame, the algorithm fails to follow him properly as he moves to the center. This might be because the color of the lecturer's shirt is very similar to the board.

c. Background Subtraction

Some rudimentary essential background subtraction has been implemented on OpenCV. It simply involves some filtering and subtraction of frames from the background frame to get a subtracted image. This subtracted image is going to be used by the Motion Tracking Module to detect the initial position of the lecturer. The main work that is going to happen over the next two weeks is going to be about improving upon the basic implementation already done.

The images given below are the sample background subtraction we have already done. It clearly mark out the lecturer, extensive filtering and other image processing techniques would have to be further applied to extend the concept to refinement.

* Simple Background Subtraction, run using the background frame (top left), on the video frame (bottom left) to give the resulting frame (subtracted) frame on the right

2. Things that have to be done

a. Finalize the modules of motion tracking and camera control.

Having achieved the goal of simple background subtraction, we now plan to apply normalization to the background subtracted image, in order to cater for the artifacts. We then need to apply motion tracking algorithms to the resulting images, so that we can detect the movement orientation of the lecturer. This will be followed by a continuous effort to improve the algorithm; since there are a lot of cases even within the limited scope of this project and the algorithm will have to cater for all of them.

As for the camera controller module, the only abstractions left are of the Zoom In and the Zoom Out functions, which are simple functions and will not take very long to implement.

b. Interface motion tracking algorithm with camera control

The immediate step after we are done with motion tracking is to create an interface between the motion tracking algorithm and the camera control module. The motion tracking algorithm will give an output indicating the direction in which the instructor is moving, and other variables like speed, distance, etc. The interfacing module will

take this data as its input, and based on that, decides whether any changes need to be made to the pan-tilt-zoom status of the camera.

This will be done first at a basic level, meaning that we will have a simple implementation, for example having the camera keep the instructor at the center of the screen throughout. Later on, when we implement our “Direction and Production Rules” module, we will make changes to the interface module so that the output video looks more professional and is closer to what a professional cameraman would make.

If you look at System Modules Hierarchy Tree (refers to Section 3 – (Aut Qtr ER) Autumn Quarter Engineering Report) of the project, it shows clearly that apart from interfacing the Motion Tracking Module with the Camera Controller Module we also have to interface these two modules with the Physical Design Rules Module (refers to 2.1.1 (d) Design Specification). The practical aspect of the Physical Design Module is to it input of the physical parameters of the lecture hall itself. As an example we’ll go through the distance parameter of the Physical Design Rules Module and how it is used by the Motion Tracking Module and the Camera Controller Module. If you enter the distance of 10 meters into the Physical Design Rules Module, the camera module should know about it, for appropriate amount of panning, tilting or zooming of the camera. The Motion Tracking Module will just give values to the Motion Tracking Module, as how many pixels the lecturer has moved. If lets say the lecturer has moved 5 pixels, the Camera Controller Module should calculate the number of degrees it has to pan or tilt (according to the 10 meters) the camera in order to keep tracking the lecturer.