Exploiting Efficiency

Programming Assignment #9
CS253 Spring 2014
Due 6:00pm Wednesday, April 27th, 2016

Motivation

Now that you have a fast program, let's exploit it by making it an inner loop!

Task

In some ways, the PA9 task simplifies the PA7/8 task. We will only use one distance measure (Euclidean distance, a.k.a. L2 distance, but normalized by dividing by the number of frames in the action), and your program will take only one Kinect video file as input. By now, however, your program should be relatively fast, and we want to exploit that.

All the long Kinect videos we have given you this semester have similar content. The person in the video performs one action (e.g. waive hands), followed by a second action (e.g. kick) and then the original action again. The task in this assignment is to identify the first action.

How long is the first action? We don't know, but we know how to compare an action (short Kinect video) to a longer Kinect video. This was assignment PA7. Now we want to find out where the first action repeats. So we need to take off the beginning of a long video and consider this the short action, and then compare it to the rest of the video. How long a prefix do we take off? We don't know, so we will try all of them. Well, almost. We assume the action has to be at least 30 frames (1 second) long. So we compare the first 30 frames of the input video to frames 31 to N, and find the match score. Then we compare the first 31 frames to frames 32 to N and find the match score, and then the first 32 frames to frames 33 to N, and so on. We stop when the prefix is as long or longer than the remaining video. When we have tested all the prefixes, we return the length of the best matching prefix.

To be clear: read in the whole video, normalize it, and then split it into prefix (action) and postfix (longer destination video). When comparing the prefix to the postfix, the distance is the Euclidean DTW distance, divided by the number of frames in the prefix.

A warnings: *don't print anything other than the start and stop positions to std::cout, or you will lose points!*

So your program takes a single long video as input, and writes the length of the first action to std::cout.

Submitting your program

You will submit your program through Canvas. You should submit a single tar file, and this file should contain all the source files of your program *and a*

makefile, but no object or executable files. The makefile should create an executable program called PA9. To grade your programs, the GTAs will write a script that untars your file in a clean directory, runs 'make', and then tests PA9 on novel inputs. If your program does not compile, whether because of an error in your makefile or an error in your code, you will receive no credit on any of the test cases. Note that as always, we will test your code on the department Linux machines.

Grading

This assignment is not graded on speed, but we will Valgrind it. It will loose points if it doesn't Valgrind cleanly.

Hints

1. Simplifying code is a good thing. You don't need all those distance measures now. Don't throw them away, but if you can simplify your code…
2. The faster your PA8 is, the easier this program will be to debug.