CS 283 Final Project Proposal

Kenny Yu, HUID: 30798260

Fall 2012

I am deciding between two projects. Feedback on the scopes of these projects would be very helpful! Also, feedback on how I should evaluate the success of these projects will be especially useful.

1 Reconstructing John Harvard from Google/Facebook Images

1.1 Motivation

Doing a Google search/Facebook search for the John Harvard statue provides many images of the statue, although with many tourists and pedestrians in the pictures. Furthermore, the statue occurs in many different lighting conditions (day/night) and weather conditions (snow, rain). In addition, the shinyness of the statue may cause some of the darker parts of the statue to not have any salient points. Thus, simply using a tool like bundler therefore may not provide robust reconstructions.

1.2 Goal

Assemble a pipeline of steps to enhance the reconstruction.

1.3 Proposed Method

Some ideas for the pipeline include:

1. Use a pedestrian detector (http://www.pedestrian-detection.com/) to either (a) throw away images with pedestrians or (b) remove salient points in regions detected to be people.

2. Segment the image based on texture in order to filter out noisy salient points.

3. Cluster images together based on scene conditions (weather, lighting), and reconstruct only from images of a certain type.

Some questions I have regarding these stages: What other useful components would be useful in this pipeline? How do we account for color variation without the raw linear color measurements? How can we measure the BRDF of a shiny surface?

Once we have filtered out the noise, I will assemble the final reconstruction using these packages:

1. Bundler - http://phototour.cs.washington.edu/bundler/
2. Clustering Views for Multi-view Stereo - http://grail.cs.washington.edu/software/cmvs/
3. Patch-based Multi-view Stereo - http://grail.cs.washington.edu/software/cmvs/

1.4 Plan for Evaluation

In order to judge the success of my project, I will judge by eye the quality of the reconstruction. One metric to judge the success will be to minimize the number of pictures needed to still obtain a good reconstruction.

2 Hand Gesture Actions

2.1 Motivation

Inspired by the Kinect and recent hand gesture technology (for an example, see http://news.cnet.com/8301-11386_3-57548636-76/bringing-minority-report-touchless-gestures-to-windows-8/), I want to explore how one can use a simple webcam to detect hand gestures and convert these into concrete actions. I also want to explore how to train a classifier to detect objects.

2.2 Goal

The goal of this project will be to use a webcam to detect simple hand gestures and convert these into concrete actions in an application, such as Google Maps or a drawing app. Examples of possible hand gestures include actions such as scrolling left/right/up/down, zooming in and out, rotating, and pointing.

2.3 Proposed Method

I will use OpenCV to perform video tracking, and I will train a classifier to detect different orientations of the hand and/or multiple hands. I will most likely use a cascade classifier to train hands: http://docs.opencv.org/doc/tutorials/objdetect/cascade_classifier/cascade_classifier.html. To train the classifier, I will use images from Google image search and or take multiple webcam pictures of my hands in different orientations.

Once I am able to accurately track the position and orientation of the hands, I will use Chrome's webRTC (http://www.webrtc.org/) capabilities to send a video stream from the client to a server that performs the video tracking and hand detection, and then convert this back into concrete actions in the client. A example of such a server-client setup is here: http://www.smartjava.org/content/face-detection-using-html5-javascript-webrtc-websockets-jetty-and-javacvopencv. An example of an application using hand gestures would be navigating Google Maps or drawing an image.

2.4 Plan for Evaluation

I will evaluate my project based on these guidelines:

1. Accuracy of the hand/finger classifier, subject to different orientations, lighting, and other noise factors.

2. Number of images required to train the classifier.

3. Number of different hand gestures that can be detected.

4. How well these different hand gestures translate into concrete actions in some application.

I will explore different kinds of classifiers, as well as different kinds of training data (hands in different orientations and lighting vs. hands in similar lighting and orientations) to test the accuracy of the classifier.

Question: What other categories should I evaluate the success of my project?