Virtual Reality Camera on Google Glass

Toan Tran, Artem Vasiliev, Vivardhan Kanoria

1. Introduction

The emerging of wearable devices like Google Glass is shifting the paradigm of how we interact with the virtual and physical worlds. With Google Glass on your eyes, the camera will see what you see. The user, hence, can link their interactions in physical world to manipulate digital objects.

We want to research and build virtual reality technologies based on computer vision that allows Google Glass users to control digital information use their hands and other physical artifacts. For the scope of this class, we research and implement two use cases.

- Using hand pointing gesture to select and highlight objects.
- Linking jewelries to photo albums and retrieving the photos later by facing the camera to the similar jewelries.

Select and Highlight Objects using Hand Pointing

When taking video or pictures, the users can point their index finger to an object in the scene to get that object highlighted, e.g. drawing a box around the selected object (Figure 1). For the scope of this project, we limit to highlighting the object after the hand has been withdrawn (i.e. hand is not in the image).

Figure 1. Assume the person (left) points at the TV (right), the TV is selected and highlighted.

Jewelry-Photo Linking and Retrieval

People take photos to capture memories of special moments. Once photos are put into albums they could be hard to find if a person want to go back to that special moment, so typically the settle for a few pictures on the desk of in the wallet. At the same time people often have artifacts associated with these special occasions, like wedding rings, military name tags, special key chains or even dedicated jewelry charm bracelets.

In this project, we want to implement a demo app that link jewelries to photo / photo albums, retrievable by facing the camera to the related jewelries. For example, wedding albums are linked to a wedding ring or tourists can recall photos of a journey when looking at souvenirs they bought from the place they visited. Hand pointing can be use to select one of many objects on the shelf.

Figure 2. Jewelry-photo linking and retrieval.

This project addresses three main problems: hand gesture detection, object selection, and image-based retrieval.

2. Selecting and Highlighting Objects using Hand Pointing

We are implementing an Bag-of-Word SVM classifier that uses SIFT features [Dardas 2011, Gurial 2012]. The workflow of the BoW SVM gesture detector consisting of 3 processing components is depicted in Figure 3.

- Skin Color and Texture Segmentation: This component classifies each pixel of the input image into either skin (1) or non-skin (0). The color and texture features are described in detail in Section 3.1. Then, connected components of skin pixels are extracted. Components that have size less than 5% of the image size are discarded because they are too small to be a meaningful hand gesture. The output of this component are windows each of which contains a connected component of skin pixels. These windows are called candidate hand windows because they may contain non-hand stuffs that have color and texture similar to those of skin.
- BoW SVM (SIFT): This is the classifier that determines if a candidate hand window contains a true hand gesture or not, and what the gesture is. This component is described in detail in Section 3.2. The output of this SVM classifier are windows that are believed to contain true hand and the type of the gesture - either pointing gesture or cropping gesture.
- Fingertip Localization: This component receives true hand windows from the SVM classifier and localizes the fingertip. The fingertip is detected by finding the top-most hand pixels because we limit our hand pointing gestures to point upward [Hagara 2013, Raheja 2011].

Figure 3. BoW SVM hand gesture detector workflow

2.1. Personalized Skin Color and Texture Calibration

Skin detectors in the literature employ single thresholds for all people. HSV-based skin detectors classify a pixel skin if its *H* value is within [0, 50] and its *S* value is within [0.23, 0.68] [Phung 2005]. Log-chromaticity based skin detectors classify a pixel skin if *log(R/G)* is within [0.15, 1.1] and *log(B/G)* is within [-4, 0.3], where *R, G, B* denote red, green, blue values of the pixel [Khanal 2011]. The use of single thresholds incurs the following separation trade-off. If the thresholding range is too small, it cannot cover all skin types. On the other hand, if the range is too big to cover all skin types, it sacrifices the discrimination capability of separating skin and non-skin stuffs. Moreover, it is difficult to adapt to skin change under different illumination conditions. From our experiments, these thresholds yield bad results on our hand collections.

This project also chooses an explicit-thresholding classifier to detect skin pixels for its fast performance, but learns different skin thresholds for different person. When a user runs the app for the first time, his/her hand on a white background is taken. This calibrated hand is used to learn the color and texture features of the user's hand skin. Our machine learning based

calibrator learns the most discriminant feature spaces for the user's skin (Figure 5). The input to the calibrator are the user's hand and 100 background images that do not contain skin.

- Skin color calibration in HSV color space: The histogram distribution of *H* and *S* values of skin pixels is first computed. Then, the calibrator searches for two optimal ranges [*minH, maxH*] and [*minS, maxS*] that best separate skin pixels from non-skin pixels.
- Skin texture calibration: Texture is computed by 2D Gabor filters. A 2D Gabor filter is a 2D Gaussian function modulated by oriented complex sinusoids. The calibrator searches for an optimal set of Gabor filter parameter, including the scale and the sinusoidal frequency, that best separate skin and non-skin pixels. In our experiment, we use four orientations for Gabor filter, including 0, 45, 90, and 135 degree.

2.2. Bag-of-Word SVM Classifier using SIFT Features

Our SVM classifier is based on the standard BoW implementation.

- Training: The first step is extracting the keypoints and SIFT features of hand gesture training images. Then, a k-means clustering algorithm is used to cluster the keypoint descriptors into K clusters and encodes each keypoint by the index of the cluster to which it belongs. Each cluster is considered as a visual word that stands for a particular local pattern shared by the keypoints in that cluster.
- Testing: First, the SIFT keypoints and descriptors of the testing image are extracted. Then, the feature descriptors are matched with the visual words created in the training step. Finally, the generated visual-word vector is fed into the SVM classifier.

2.3. Object Segmentation and Selection

General object segmentation and selection are extremely hard. There are so much variability in real world data, and oftentimes semantic interpretation is required to correctly identify something as an object. State-of-the-art object selection algorithms either relax the problem or use a lot of training data. For the scope of this project, we will limit our object to human and employ state-of-the-art human detection techniques with modifications and optimizations for mobile app.

3. Jewelry-Photo Linking and Retrieval

We plan to implement this app based on the techniques and designs described in [Girod 2011]. Specifically, we will represent the images using Bag-of-feature SIFT or SURF descriptors. For the scope of this project, we will save the database on the mobile device and perform all image-processing processes on the mobile device, including feature extraction, feature matching, and image retrieval (Figure 4).

Figure 4. Jewelry-photo linking and retrieval app design.

4. Target Platforms

We want to implement this technology for Google Glass. However, we cannot guarantee that we will

able to make a demo on Glass for the following two reasons. First, we may not able to find a Glass. Second, suppose we can access to a Glass, the current hardware and software configuration of the Glass may not able to support this application, instead require some complicated hack that is beyond the scope of this class project. In either case, we will make a demo on Android smartphone.

5. Milestones

- May 18 to May 25: Finish hand pointing gesture detection.
- May 26 to June 01:
  - Finish object (human) selection and highlighting.
  - Finish jewelry-photo linking and retrieval.
- June 02 to June 08: Finish presentation and final report.

Reference

[Dardas 2011] N.H. Dardas and N.D. Georganas. Real-time Hand Gesture Detection and Recognition using Bag-of- Features and Support Vector Machine Techniques. IEEE Transactions On Instrumentation And Measurement, 2011.

[Gurjal 2012] P. Gurjal and K.Kunnur. Real-Time Hand Gesture Recognition Using SIFT. International Journal of Electronics and Electrical Engineering, Volume 2 Issue 3 (March 2012)

[Khanal 2011] B. Khanal and D. Sidibé. Efficient Skin Detection under Severe Illumination Changes and Shadows, In proceeding of: Intelligent Robotics and Applications - 4th International Conference, ICIRA 2011.

[Sarkar 2013] A.R. Sarkar, G. Sanyal, and S. Majumder. Hand Gesture Recognition Systems: A Survey. International Journal of Computer Applications (0975 – 8887), Volume 71– No.15, May 2013

[Phung 2005] S. L. Phung, A. Bouzerdoum, and D. Chai, "Skin Segmentation Using Color Pixel Classification: Analysis and Comparison", IEEE transactions on pattern analysis and machine intelligence, vol. 27, no. 1, pp. 148-154, 2005.

[Hagara 2013] M. Hagara. Fingertip Detection for Virtual Keyboard Based on Camera. 23th Conference Radioelektronika 2013, April 16-17, 356-360.

[Raheja 2011] J. Raheja. An Efficient Real Time Method of Fingertip Detection. International Conference on "Trends in Industrial Measurements and Automation" TIMA–2011

[Girod 2011] B. Girod, V. Chandrasekhar, D. M. Chen, N. M. Cheung, R. Grzeszczuk, Y. Reznik, G. Takacs, S. S. Tsai and R. Vedantham, "Mobile Visual Search", *IEEE Signal Processing Magazine*, vol. 28, no. 4, pp. 61-76, July 2011.