Gesture-Based Paint

Angus Ding (ad3180)
Ayaka Kume (ak3682)

1. Introduction

In this experiment, we will be implementing a system that allows the user to paint on the screen using intuitive gestures instead of the mouse. Instead of using any specific draw pad or other device, we will implement the system using a simple white paper, a webcam, and a light source. Without any tactile input, we are going to rely on visual inputs to determine the gesture and the position of user's hand in real-time. The challenging part about this project is how to define the natural gestures that human use to indicate the "drawing" on a blank paper, and how to recognize them using pure visual signal processing. Because the system only rely on the visual input, this project is perfectly suitable as an example of a visual interface.

2. Previous Work

EnhancedDesk [1] is a two handed drawing system using infrared camera. Left hand and right hand have the different role. Isard, Michael, and John MacCormick implemented a vision based drawing package to demonstrate the hand tracking method [2].

3. Program Features

The user will be given a device that consists of a white paper(or a whiteboard), a webcam that looks down from above, and a light source that projects light onto the paper from a non-perpendicular angle. The distance between the webcam and the paper, the paper and the light source, and the angle of the light source are all fixed. On the bottom of the paper will be some color blocks which represent a palette, and possibly some symbols which represent the drawing tools that the user can choose. The user can simply touch the color blocks to choose the color, and touch the tool symbols to choose the painting tool he or she wants to use. On the same time there will be a program on the computer screen which shows the canvas on which the user draws.

To draw a picture, the user can use the most intuitive gesture --- use the index finger like a pen. Touching the paper with the index finger means to draw, while moving the finger without touching the paper means to move the pen without drawing.

This gesture, when the user touches the palette or the symbols instead of the empty area, means to select the color or the tool instead of drawing.

For convenience, we will also define an ‘erase’ gesture, which is a palm facing downwards with the four fingers stretching straight. This gesture is easy to use, and suitable for the semantic of erase, because it is the movement one will use to wipe something away from a surface.

We will also define a drag gesture which is the ‘victory’ gesture. One of the tools that we supply is the “selection” tool, which allows the user to select an area on the canvas. Then the user can “drag” the selected area around with the drag gesture.

4. Implementation

The program will analyze each frame taken by the webcam to determine the user’s intention. The first step for each frame is to determine which gesture the user is posing now. For the three gestures defined above, we then have to determine the position of the fingertip(or in the case of the palm, the front end of the palm), because in our grammar it represents the actual location where the user wants to paint, erase, or do other command. We also have to determine whether the user’s hand touches the paper or not, for it indicates whether the user wants to actually paint on the canvas, or is just moving the hand around.

To recognize a hand gesture is essentially classifying the image into one of the predefined classes. This is a typical classification problem in machine learning. As such, we will use machine learning technique to solve this problem. The specific classifier we will use is the simple nearest neighbor classifier. To extract feature, we first find the region of the hand using skin color segmentation. This should be easy since the background is pure white. Then, using the method proposed by [3], we will compute the orientation histogram of the hand region, and use it as the feature of the frame. The advantage of the orientation histogram is that it is irrelevant to the position of the hand, and it is easy to deal with the different orientations of the hand. If two frames contain the same hand gesture but with different orientations, their orientation histograms will have the similar shape, but will have a different disposition. We can find the orientation of the major axis of the hand region and translate the histogram accordingly.

Then we have to find the position of the fingertip. We are going to use the method proposed by [5] to identify the wrist end and also to crop the hand. Then, we will find the center of mass of the hand. For the gestures we defined, we can simply define the fingertip to be the pixel that is farthest from the center of mass of the hand, and is on the opposite side of the wrist.

To determine if the user's fingertip touches the paper, it will be hard to only use the information of the hand since the camera is looking from above. To solve this problem, we add a light source that projects light from some inclined angle. The shadow of the hand projected by this light source will thus have a horizontal disposition proportional to the height of the hand. Thus, the position of the fingertip of the shadow will overlap with the real fingertip if and only if the finger touches the paper. To make sure that the hand doesn't block the shadow, the light source should be placed at some place that is roughly perpendicular to the arm. To find the shadow fingertip, we can assume that the wrist end of the shadow is the same as the real hand. Because of the distortion caused by the projection, tough, the fingertip of the shadow may not be the pixel that is farthest from the center of the shadow hand. However, we can still set it to be the shadow pixel that is farthest from the wrist end.

This completes the interface we are going to use. The rest of the job is to implement the paint program itself. We will ignore the details of that here since that is not related to the topic of this course.

5. Evaluation Metrics

We will evaluate the system in four ways. First, we will evaluate the gesture recognition accuracy. We will be testing with at least 60 training data, 15 for each defined posture and 15 for undefined posture. The accuracy is calculated as the score divided by the number of trials for that symbol. For the trial for unknown gestures, the program gets 1 score if it outputs 'unknown', and the accuracy is computed similarly. Thus we will have 8 accuracies along with 1 overall accuracy. To make the system robust, all accuracies should be at least 90%.

Then we will evaluate the overall system in three different views, accuracy, consistency, and speed using user study. For accuracy, we ask an user to trace dots and circles and measure the accuracy of them. The accuracy of a dot can be represented as the distance between the point which the user draws and the target shown to the user. The accuracy of a circle can be represented as the average absolute difference of the distance from the center of the circle. For consistency, we ask user to draw a dot (or a circle) at the same place several times and measure the standard deviation of each trial. For time, we measure the time an user draw a task, which starts from when the task appears on the screen and ends when an user finished drawing and press 'q' in the keyboard.

There are two ways to evaluate our system in terms of the measurement defined above. One way is to have one of us practice this system for a sufficient amount of time and to use his accuracy/consistency/time as a baseline. We can then see how a first-time user does compared to the baseline to determine how easy it is to use the system. Another way is to compare the results of the same users when they use our system with that when they use a similar moused based program, such as MS Paint.

Tasks Assignment

Task	Assignee	Deadline
Project Proposal	Angus	4/31
Posture Recognition	Angus	4/13
Fingertip detection	Ayaka	5/8
Contact detection	Angus	5/7
GUI	Angus	5/9
Evaluation	Ayaka	5/11
Documentation	Both	5/1?

Reference

[1] Chen, Xinlei, et al. "Two-handed drawing on augmented desk system." *Proceedings of the Working Conference on Advanced Visual Interfaces*. ACM, 2002.

[2]Isard, Michael, and John MacCormick. *Hand tracking for vision-based drawing*. Technical report, Visual Dynamics Group, Department of Engineering Science, University of Oxford, 2000.

[3]Freeman, William T., and Michal Roth. "Orientation histograms for hand gesture recognition." *International workshop on automatic face and gesture recognition*. Vol. 12. 1995.

[4]Chen, Qing, Nicolas D. Georganas, and Emil M. Petriu. "Real-time vision-based hand gesture recognition using haar-like features." *Instrumentation and Measurement Technology Conference Proceedings, 2007. IMTC 2007. IEEE*. IEEE, 2007.

[5] Raheja, Jagdish Lal, Karen Das, and Ankit Chaudhary. "An efficient real time method of fingertip detection." *arXiv preprint arXiv:1108.0502* (2011).