Slides Real-Time Remote Hand Control with Ada-Boost and Haar Features

Xie Zhe, Zhao Chen

Abstract

Slides are widely used in presentations to make an understandable demonstration. However, switching the slides by walking to the in front of the computer and pressing page down could be seen as an interruption for both the lecturer and audience. To solve this problem we have developed a slide control application to page down/up the slides by recognizing the hand (fist) movement of the lecturer captured by the camera. The algorithm is conducted under the object detection framework proposed by Paul Viola and Michael Jones, which is a rapid object detection algorithm with high detection rate and accuracy. The performance in the real life testing nicely meets its intended mission.

Key words: OpenCV implementation, Haar feature, AdaBoost

Introduction

With the help of slide tools such as PowerPoint and adobe reader, nowadays the lecturer could present much more multi-media contents for the audience in the presentation. One inconvenience during the process is that the lecturer has to walk to the keyboard to press the arrow key or space key or enter key to switch to the next page. Even the laser pointer with page down/up buttons are invented, it may not likely to become a standard configuration of the computer for its inconvenience.

The web-camera becomes more and more popular in the laptops, making it possible to access to the lecturer's image any time. There are some software products available in the market. Tencent Gesture PPT Controller [Tencent, 2011] is a lab-product that use Haar feature [Viola, 2011] under the name of one of the largest internet company in China. It has not become widely accepted by the public mainly because it cannot recognize any gesture one meter away from the camera. The most recent product in the industry is proposed with the help of Kinect [Blake, 2012]. Kinect is a motion sensing input device developed by Microsoft.

In our application, we try to not introduce any other external hardware except for the web-camera. Two algorithms are involved in our application: the Haar feature which is responsible for the hand detection; the AdaBoost [Freund, 1995] for promotions of recognition process. These are discussed in the next section. We use the framework provided by OpenCV which is illustrated in the third section and followed by the discuss and conclusions.

Algorithms

There are three highlights in this proposed object detection application. The first is the integrated image used for rapidly computing the Haar features for images captured by the camera. The second is the AdaBoost algorithm used to select the important features among thousands of Haar features and the third one is a method of cascading [Viola, 2001] multiple classifiers to accelerate the detection speed. More details will be given in the following:

According to the framework, images are classified by the feature values rather than the pixel values. The features used are called Haar features or rectangular features. There are three kinds of Haar features: two-rectangle feature, three-rectangle feature and the four-rectangle feature. The value of a two-rectangle feature is the difference of the summation of the pixel values in two rectangle regions. The value of a three-rectangle feature is the summation of the pixel values in the outside rectangular regions subtracted from the sum in the center region and finally the four-rectangle feature just calculate the difference between the two diagonal parts.

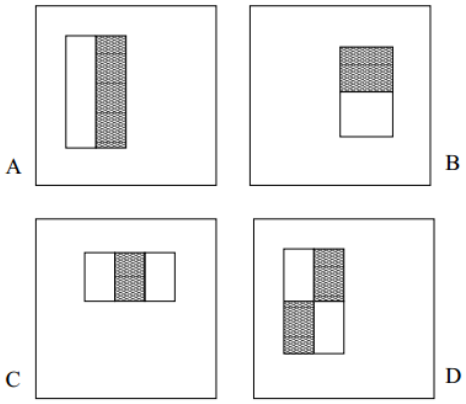

Figure 1: Example rectangle features [Viola, 2001]

The main advantage of the Haar feature is its efficiency in computation by using the integral image.

An integral image is just another way to present the original image, in which a pixel at location (x, y) contains the summation of all the pixels above and to left of the (x, y) in the original image. Then after the computation of the integral image, any rectangular sum can be computed with four points in the integral image. And more specifically, the two-rectangle feature can be computed in six references in the integral image and eight for three-rectangle features and nine for four-rectangle feature.

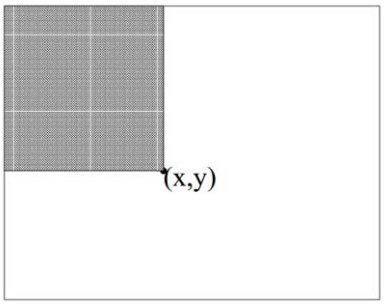

Figure 2: The value of the integral image at point (x, y) is the sum of all the pixels above and to the left. [Viola, 2001]

There are 45,396 rectangle features in each sub-window [Viola, 2001] and clearly we can't computer all of them. So we will apply the AdaBoost learning algorithm to select the most distinctive ones among those features. AdaBoost learning algorithm is shown in the figure below:

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i \left| h_j(x_i) - y_i \right|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
  where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
where $\alpha_t = \log \frac{1}{\beta_t}$

Figure 3: the Adaboost algorithm for selecting features [Viola, 2001]

Via the AdaBoost algorithm, we can select a set of classifiers: some of them are simple thus can be rapidly computed, however, with weak classification ability and some are complex and strong classifiers but heavily

time-demanding. In order to make the detecting process very efficient and rapid, we will introduce the procedure of cascading: we will cascade all the classifiers from the simplest ones to the most complicated ones. In this way, we can just reject the majority of the candidates with the simple classifiers and leave the most confusing ones to the complex classifiers.

Application Framework

The whole application is originally programmed with C++ under the existing framework OpenCV [Itseez, 2013]. OpenCV is an open source computer vision library first provided by Intel Cooperation and then by the Developers Team “Itseez”. After implementing the functions in C++, we rewrite the code to C# with a user interface under Visual studio 2010.

The detection with Haar feature and AdaBoost algorithm is integrated in the OpenCV machine learning library as the “DetectHaarCascade” function with given image and cascade.

To avoid the long proceeding time of the Haar detection and make the application become real time even the user is far away from the camera, we first only try to detect the location of the lecturer’s body. Then the time consuming hand detection procedure will be preceded only in a small limited area. Thus, the movement can be detected in real time.

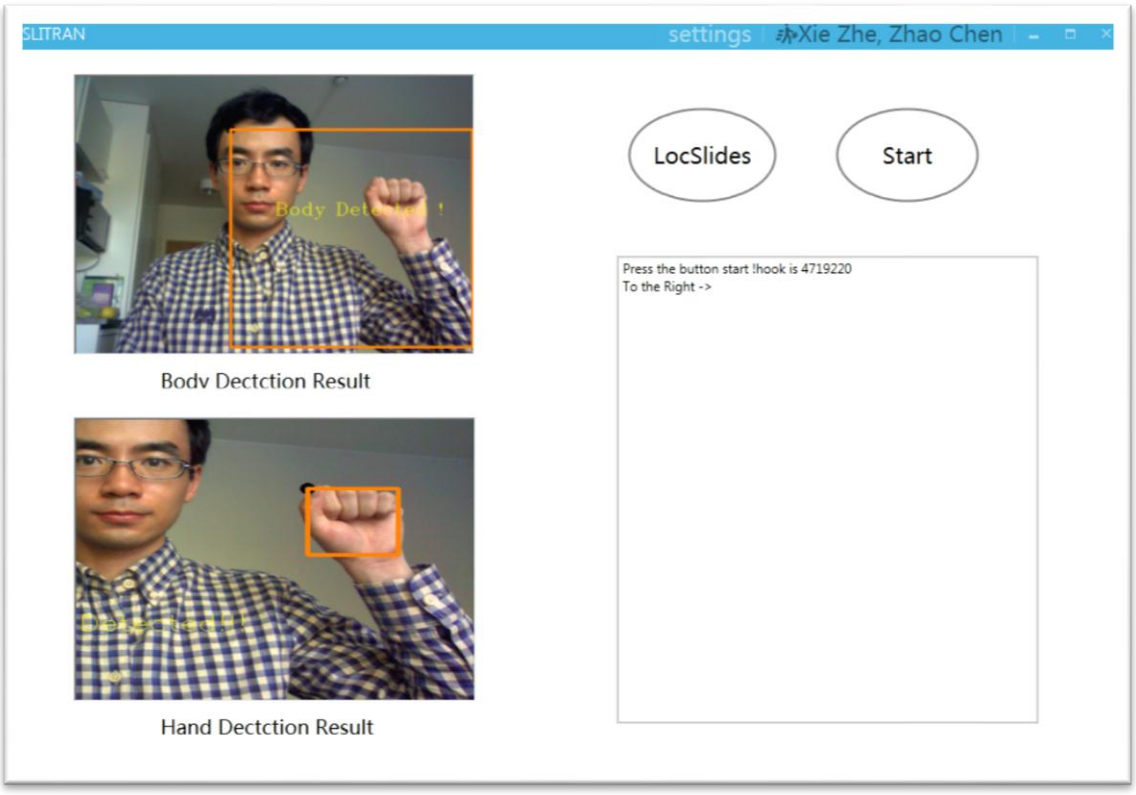

Figure 4: The application’s user interface

There are three sub windows in the application. The one in the top left shows the detected location of the body. The bottom left shows the detected location of hand (only fist here). The right window shows the movement recognized by the software with the “start” button and the button for testing if a PDF file is opening.

Discuss and Conclusion

Although detecting the fist movement in real time with high accuracy is realized, there are several limitations. The most important one is that, since the training for the Haar feature requires a huge database, we simply use a cascade from the Internet. Thus, it limits the user have to move his fist while keeping the upright position. In the presentation it may look a little bit strange. Our next step is to train our own hand cascade. Apart from that, we may make the program adaptable to more environments (it only supports Windows now) and more functions such as start/end full screen status. Another aspect for improvement is about the documentation and the redesign of user interface to make the code easier to understand and the application more convenient to use.

Above all, this demo is a success by meeting the requirements of real time long distance object (fist) recognition. The combination of Haar feature and AdaBoost algorithm provide fast image processing, meanwhile the well-trained cascade leads to an acceptable recognition rate.

Reference

P.A. Viola and M.J. Jones, "Robust Real-Time Face Detection",; in Proc. ICCV, 2001.

Yoav Freund and Robert E. Schapire, " A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting", 1995.

P.A. Viola and M.J. Jones, "Robust Real-Time Object Detection", International Journal of Computer Vision, 2001.

Y. Chen, M. Liu, J. Liu, Z. Shen, and W. Pan, "Slideshow: Gesture-aware PPT presentation", in Proc. ICME, 2011, pp.1-4.

Joshua Blake, "Kinect PowerPoint Control v1.1", http://kinectpowerpoint.codeplex.com, 2012

Tencent Company, "Tencent Gesture PPT Controller", http://labs.qq.com/product.php?id=1, 2011

Itseez, "Open Source Computer Vision Library", Intel Corporation, http://opencv.org, 2013