TSB-funded Project 'TADD' - Trainable vision-based anomaly detection and diagnosis Technical Report for January 2014

Ran Song and Tom Duckett

Agri-Food Technology Research Group, Lincoln Centre for Autonomous Systems Research, School of Computer Science, University of Lincoln, UK

Abstract

In December 2013, we reported an automatic method in order to detect the region of a label within a tray image, given that that the tray image without the label has been provided. In this report, we extend the method to solve the following problem: how can we automatically and robustly evaluate the position and the orientation errors of a label within any tray image? Preliminary experiments show that the proposed method is efficient, reliable and easy-to-use (in particular for non-experts) while certainly more tests need to be done to comprehensively evaluate it.

1. Introduction

In this report, we describe our newly developed method for automatically detecting the position error and the orientation error of a label within a test image of a food tray, given that two reference images have been provided. The first reference image is a tray image without any label and the second one is a tray image with the label at the right position. The basic idea is to teach the system by first inputting the two reference images and then using the learned knowledge to compute the errors of the test image with regard to the reference images.

2. Method

The method contains three major steps. First, we register the second reference image to the first reference image and detect the rectangular label

Figure 1: The three input images. Top: the first reference image; Middle: the second reference image; Bottom: the test image

region using the technique that we proposed in our last monthly report. Second, we employ the same technique to register the test image to the first reference image and also detect another rectangular label region. The third step is to measure the positional difference between the two rectangular regions, defined by the so-called 'position error' and 'orientation error'.

We here just simply review the technique for registration and label region detection. It includes three components:

1. Feature extraction using the Speeded Up Robust Features (SURF) algorithm [1];
2. Feature matching and image registration using the Random Sample Consensus method (RANSAC) [2] and the Levenberg-Mardquardt algorithm;
3. Label region localisation and repairing.

Note that the method proposed in our last report is designed to deal with the images captured by a hand-held camera where their backgrounds are black. There are some changes in the latest version of our software to make sure that it can handle the images (for example, we show the three input images in Figure 1) captured by the camera mounted on the real Ishida machine in Witham Wharf in Lincoln.

1. Set the intensity values of the pixels around the border region of the registered images to $I(500, 1) - 10$ where $I(500, 1)$ represents the intensity value of the pixel with row number=500 and column number=1 in the first reference image. In the previous version, the values are set to 0 by default.
2. Give a stronger blurring to the difference image computed by subtracting the registered images. In the previous version, we blur every pixel with its neighbours in a 10×10 neighbourhood while in the latest version, we use a 40×40 one.
3. Adjust the threshold by 25% which converts the difference image from an intensity image to a binary image.

With the help of these changes, we solve the problem caused by the inconsistent background illustrated in Figure 2. And this leads to the successful detection shown in Figure 3. In Figure 3, the yellow rectangle denotes the position of the label in the test image and the blue one denotes where it should be (the correct position). Our software also outputs a list of statistics as shown in Figure 4, which can be used to quantitatively evaluate whether the position of the label in the test image is wrong and how wrong it is.

Figure 2: The inconsistency within the background region (in particular, the border regions pointed by the red arrows) is compensated by resetting the pixels values.

3. More results

More implementation results are shown in Figures 5 and 6 where the top rows show the input images and the bottom rows show the detection results. In particular, in Figure 5, we input a test image which is exactly the same with the second reference image. As we expect, the position and the orientation errors are both equal to 0 and the two rectangles are overlapped with each other.

4. Future work

One limitation for the current software is that setting the pixel value to a specific one, such as $I(500, 1) - 10$ that we used in this work, is not reliable. There would be a larger possibility for the software to fail if the rotation angel is too large since it would produce larger inconsistent border regions (please refer to Figure 2 for a better understanding). Thus currently, we assume that all rotation angles are smaller than 45 degree although it is highly likely to be so in practice. A reliable solution is to change the

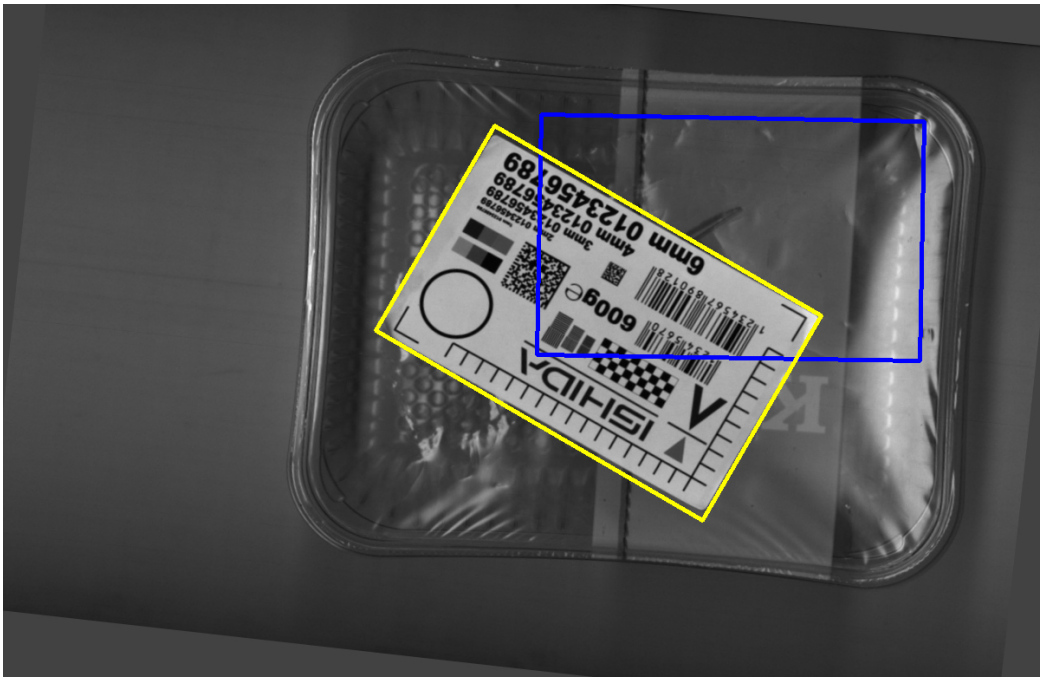

Figure 3: Visualisation of label region detection.

conveyor belt of the Ishida machine to a black one. Secondly, the current software requires manual input. The future work will try to get it connected with the Ishida machine with running conveyor and automatically read the input images captured by the camera mounted on the machine. This seems not trivial. Simply checking the time stamp of the image saved on the disk does not work since its precision is just minutes (it could be seconds if we use some external library but it is still not enough). Within one minute or one second, the image captured by the camera and then saved on the disk could be modified since the camera has already taken another shot. Also, to use such a strategy, the software needs to frequently check the time stamp of the image, it will slow down the processing. A reliable solution is to look up the software attached to the camera to check whether a certain flag (a parameter in BOOL type) is changed (e.g., from 0 to 1) whenever it takes a shot. It is known that for a webcam, such a flag does exist and it can be called within the framework of OpenCV.

```
C:\Users\computing\Documents\Visual Studio 2010\Projects\imagerega\Debug>imagere
g img0000.jpg img0001.jpg img0002.jpg
Centre point: [839.396, 269.404]
Orientation: -89.0317
Centre point: [686.884, 367.722]
Orientation: -59.8935
Position error: 181.457
Orientation error: 29.1382
```

Figure 4: The statistics of label detection. The 'Centre point' displays the coordinates of the centre point of the label and the 'Orientation' displays its orientation angel. The 'Position error' is calculated by the distance between the two centre points and the 'Orientation error' is measured by the included angle between the two orientation angles.

Figure 5: The results of label detection

References

[1] H. Bay, T. Tuytelaars, L. Van Gool, Surf: Speeded up robust features, in: Proc. ECCV, Springer, 2006, pp. 404–417.

[2] M.A. Fischler, R.C. Bolles, Random sample consensus: a paradigm for

Figure 6: The results of label detection

model fitting with applications to image analysis and automated cartography, Communications of the ACM 24 (1981) 381–395.