Monthly Report– February 2014

Hossein Malekmohamadi

February 28, 2014

Contents

List of Figures

Chapter 1

Introduction

This is the February 2014 report for the project Trainable Vision-based Anomaly Detection and Diagnosis (TADD). The aim of this project is to deliver a software system capable of detecting and recognising intelligently different anomalies in food contents and packages, e.g. potato or food container. Currently, this system can do semantic segmentation, bar-code region detection and foreground extraction based on some image processing techniques and the Adaboost learning. These features can be used in different parts of QC which is an industrial aim in TADD. However, further improvements to TADD can be accomplished by reinforcing semantic segmentation, applying 3D/UV imaging apparatus and enhancing learning capabilities. In the November 2013 report, we gave a brief introduction to a possible 3D vision system and decision forests. In the December 2013 report, we created training and test datasets for semantic segmentation for further research based on multiple datasets. In the January 2014 report, the aims were investigating 3D recording devices and extracting 3D features to analyse food contents. However, with the advent of the updated requirements after the quarterly meeting in the early February 2014, the new priorities for this quarter are as follow:

- Creating the Github repositories for both the current TADD in UoL and SB-TADD.
- Investigating HALCON capabilities for this project
- Investigating open-source OCR libraries for this project and integrate it to our current Label-TADD.
- Making a universal TADD w.r.t to current TADD in UoL and SB-TADD

In this report, we cover all topics above.

Chapter 2

TADD Software Developments

For a universal approach in software development of TADD, we needed to create repositories for previous software versions. Consequently, we came to this point to upload an document UoL-TADD and SB-TADD in Github and keep them as private repositories in LCAS-UoL members. The url for UoL-TADD (TADD-V1) and SB-TADD (TADD-V2) is: https://github.com/LCAS/TADD-project. The readme file for this project states that, the required Libraries are:

1. OpenCV 2.3.1
2. Qt 4.7.3
3. Boost 1.47
4. TBB (Intel Threading Building Blocks) $tbb40_20120408oss$
5. CUDA 4.0 (will build with 4.1)

TADD modules are:

1. QtTODD (main, startup project)
   (a) QtCvImageWidget (paints capture frame to widget)
   (b) QtCvOverlageWidget (paints overlay and accepts input)
   (c) TODDengine (90% of processes are done here)
      i. CvCamera (wrapper for different capture methods)
         A. CvCapture (capture from standard webcam)
         B. videoInput (library used to try control exposure and gain)

C. CvAviCapture (using pre-recorded as input)
D. CvFlyCapture (point grey camera 640x 480)

ii. cSLIC (CPU parts of SLIC algorithm)
A. gSLIC(GPU parts of SLIC algorithm)

iii. MultiAdaboost (Adaboost algorithm, training and learning)

And finally new items for TADD-V2 are:

1. User can assign class labels
2. New load and save features for classifier
3. Zoom to select feature in UI
4. Sliding window in UI
5. HD-Camera API integration in EDSDK library [1]

As stated earlier, one of the main objectives for this current quarter is to unify software versions and have a QA-TADD with proper documentations and recommendations to be installed in any PC with a GPU. Moreover, we have started hardware setup with the help of a technician to mount a high resolution camera to SB-TADD rather than an off the shelf webcam for quality assessment of potatoes. We have started to replace OpenCV 2.3.1 with the latest version of OpenCV (2.4.8) because there are some feature extraction techniques which can be found in them, e.g. Local binary patterns (LBP). Other superpixel features that we can include are: 1) colour histogram, 2) dilated colour histogram, 3) colour thumbnails or 4) sift clusters [2]. Finally, our aim is to cover QA needs of our project partners as much as possible. One example of the QA report is shown in Figure 2.1. A generic specification for QA can include:

1. Reject at 5% critical defects (pest damage, bruising, rots, internal defects e.g. Hollow Heart)
2. Reject at 10% mechanical damage
3. Reject at 30% skin finish, greens, (visual defects)

The above percentages are a guide for the Intake QA team in Branston ltd and can be challenged and if needed. The dimensions of the tray used in QA are:

- Height: 70mm

- Length: 600mm
- Width: 400mm

Branston Lincoln **Quality Report**

Sample Type	Intake	Forward	Routine	Cold Store	Hot Box	Store Loading

Stock assessed as suitable for:

Tesco	Waitrose	Booker	Value	Reject

Grower	R Lindley Ltd	Date Assessed	24/02/2014
Variety	Melody	QA	J.Venkova

Recommend packing into:

Tesco	Waitrose	Booker	Value	Reject

Field Ref	Alan Wilson	Consignment no.	147225

Total pick-off 25 %

Store Ref		Sample Temp °C	10
Box ID		Dry Matter %	22.2

Size Analysis			Defects	Weight	Critical Defect Wt	
Whites	Weight	%	Bruising		0.5	Total wt
< 45mm		0	Rots			0.5
45-65	11.755	65	Pest Damage			3%
65-85	6.32	35	Mech. Damage		0.645	
		0	Greens		0.86	
85+		0	Sprouting			
Total Wt	18.08	kg	Skin Finish		2.435	
Tesco Wt	18.08	kg	Growth Cracks			
				Total weight of defects	4.44	kg

Comments

Internal Defects (10 tubers)

None	Minor	Severe
10	0	0

Skin Finish Key

1 = Level 1	2 = Level 2
3 = Level 3	4 = Level 4
Scab	1 to 2
Silver Scurf	3 to 4
Black Scurf	
Skin Spot	
Netting	3 to 4

Figure 2.1: Quality report sample from Branston ltd

Chapter 3

Label-TADD: OCR integration

In this chapter we have used Tesseract-OCR [3] to read characters in digital images using C++ programming. This library has Apache License v2.0 [4]. For this purpose, we gathered data from multiple situations, e.g. date, label,... Figures 3.1 to 3.6 show the results for the original captured image, filtered image (denoised and threshold truncated image) and resized image (double size). For filtering, we have used OpenCV functions in C++ environment.

Figure 3.1: OCR results for the original image, filtered image and resized image for a part of a label (printed characters are all numbers)

Meanwhile, we have investigate MVTec HALCON [5]. This software is not open source but it outperforms in cases like Figure 3.7 when we have a dot-printed character on a graphical label. This example is taken from HALCON demo software. we have used this digital image to test the open source OCR as shown in Figure 3.8 where it shows that we need to put effort

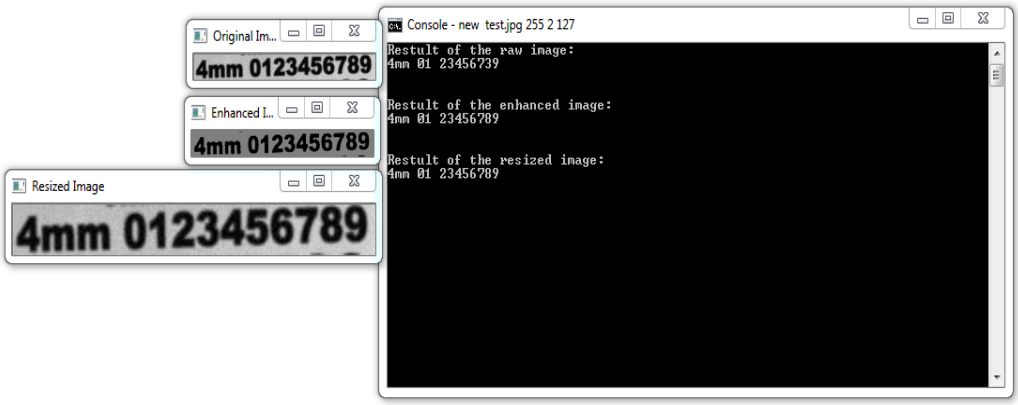

Figure 3.2: OCR results for the original image, filtered image and resized image for a part of a label (printed characters are letters and numbers)

on image enhancement techniques, e.g. background separation. In the demo version, there are few more interesting use cases like: photometric stereo in shampoo container defect detection as shown in Figure 3.9 or leather surface investigation which can be converted and used in TADD applications for quality inspections.

Finally, another problem in using OCR is how to detect text regions. We need to develop a package in C++ for this purpose and divide the captured image to multiple sub-images containing text. Then we feed each sub-image to OCR. An example of ideal text localisation is shown in Figure 3.10. We show other examples of text localisation based on [6] in Figures 3.11 to 3.14. A very good resource for this challenge can be found in [7, 8]. Another example of web-based text localisation based on [8] is shown in Figures 3.15 and 3.16.

Having said these, the final suggestion for Label-TADD is:

1. Detect label region an compare it with reference image
2. Reject packs with position/orientation errors more than a threshold
3. For passed data, perform text localisation and output sub-images containing text
4. Perform image enhancement on each sub-image
5. OCR on each filtered sub-image

Figure 3.3: OCR results for the original image, filtered image and resized image for a part of a label (printed characters are letters)

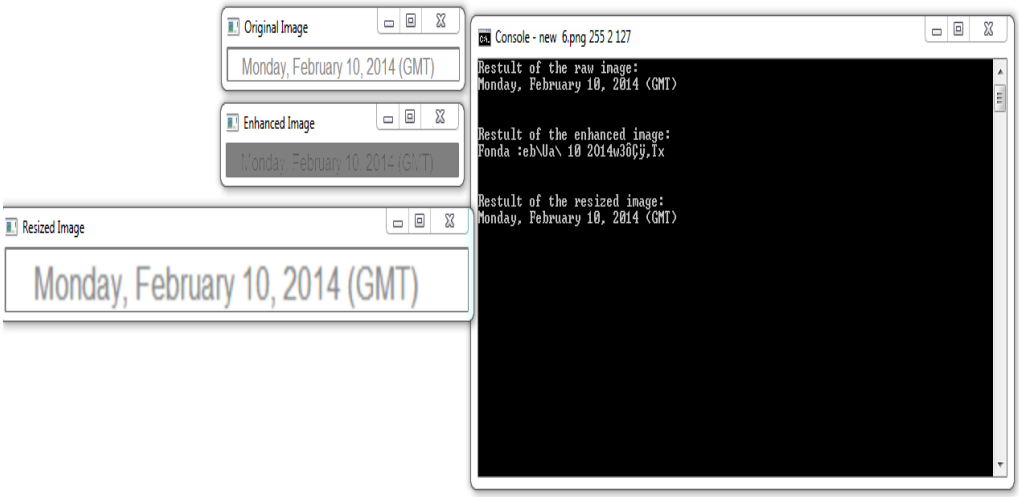

Figure 3.4: OCR results for the original image, filtered image and resized image for a part of a label (date is printed)

Figure 3.5: OCR results for the original image, filtered image and resized image for a part of a label (a brand name)

Figure 3.6: OCR results for the original image, filtered image and resized image for a part of a label (multiple lines of alphabets and numbers)

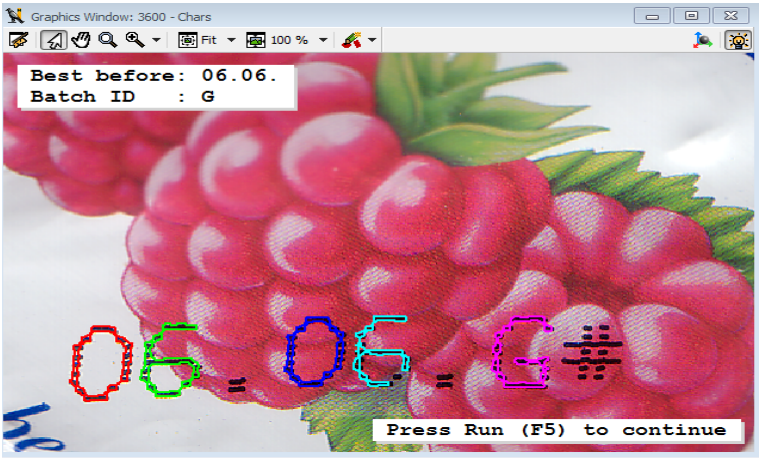

Figure 3.7: OCR demo of HALCON for a dot-printed date on a graphical background

Figure 3.8: Tesseract-OCR for a dot-printed date on a graphical background. It shows that for this kind of OCR application we need to put effort on image enhancement techniques, e.g. background segmentation or image morphology for the dot-printed area

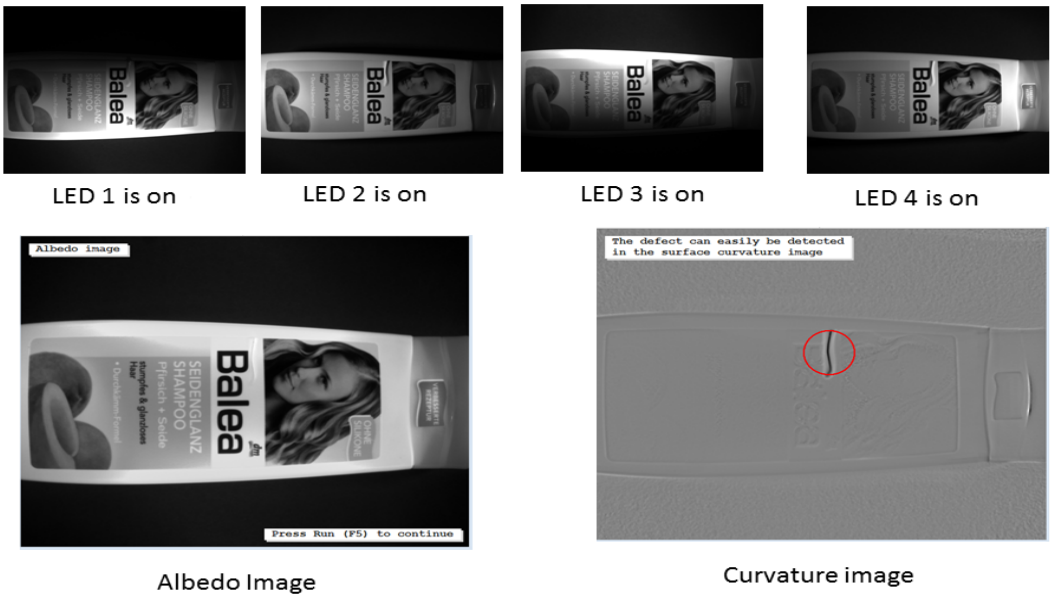

Figure 3.9: Demo of photometric application in HALCON to detect surface defect in a shampoo container

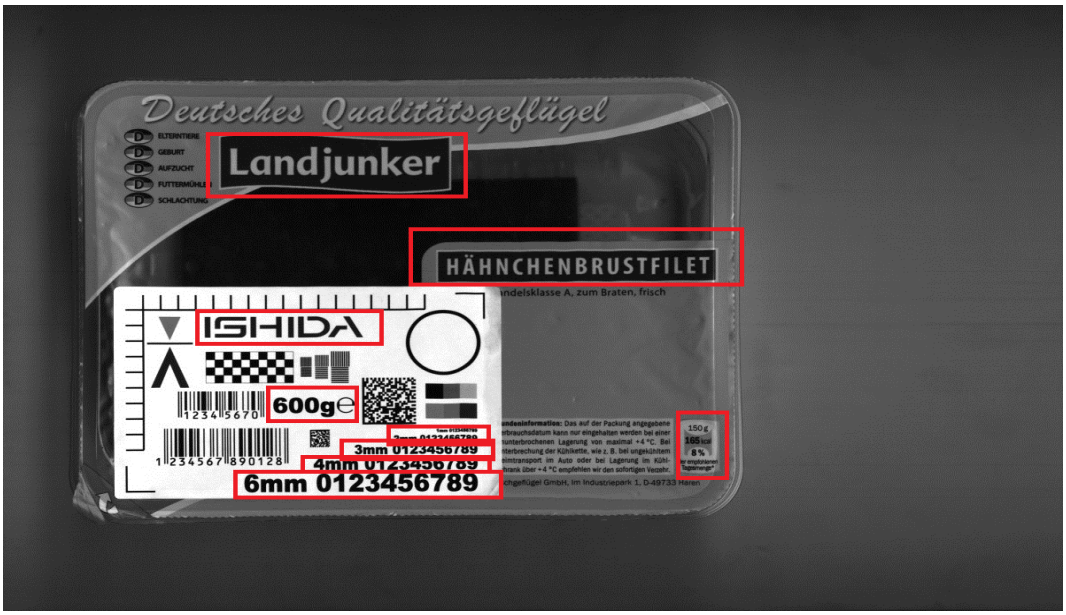

Figure 3.10: An ideal text localisation where text regions are marked. We need an automated procedure to detect these regions and create sub-images from them before OCR.

Figure 3.11: Text regions are detected using [6].

Figure 3.12: Text regions are detected using [6]. Comparing this example with Figure 3.11 shows that we need a robust text localisation algorithm that is not subject to small orientations.

Figure 3.13: Text regions are detected using [6]. There are some false detections in the barcode region.

Figure 3.14: OCR results for the original image, filtered image and resized image for the output of Figure 3.13.

Figure 3.15: OCR results for a captured image applying web-based text localisation in [8].

Figure 3.16: OCR results for a captured image applying web-based text localisation in [8]. The algorithm in [8] shows robustness to orientation errors within a tolerance.

Chapter 4

Conclusions

In this February 2014 report, the covered objectives in the TADD projects are:

- Attending quarterly review meeting in UoL and visiting Branston ltd in Lincoln.
- Software development stage for both current TADD and SB-TADD including setting up repository, installing new high resolution camera and documenting both projects.
- Testing open source OCR on several captured images using the Ishida machine in UoL, this part of the project is ready for integration to the Label-TADD.
- Investigating HALCON capabilities for the future plan of the project. Where we came across to the point that HALCON outperform open source softwares in some instances like dot-printed OCR on graphical background, e.g. printed date on yoghurt lid.
- Investigating some pre-existing techniques for text localisation to be implemented in the Label-TADD.

Bibliography

[1] http://www.usa.canon.com/cusa/consumer/standard_display/sdk_homepage

[2] Tighe, Joseph, and Svetlana Lazebnik. "Superparsing: scalable nonparametric image parsing with superpixels." Computer VisionECCV 2010. Springer Berlin Heidelberg, 2010. 352-365.

[3] https://code.google.com/p/tesseract-ocr

[4] http://www.apache.org/licenses/LICENSE-2.0

[5] http://www.halcon.com/

[6] Neumann, Lukas, and Jiri Matas. "Real-time scene text localization and recognition." Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012.

[7] http://cmp.felk.cvut.cz/ neumalu1/

[8] http://www.textspotter.org/