TSB-funded Project 'TADD' - Trainable vision-based anomaly detection and diagnosis

Ran Song and Tom Duckett

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

University of Lincoln, UK

Technical Quarterly Report

Nov 2013

Contents

List of Figures

Chapter 1

Introduction

The 'Trainable Vision-based Anomaly Detection and Diagnosis' (TADD) project aims at developing a system, delivered as software package, which can automatically, intelligently and efficiently detect and recognise anomalies on the considered objects (particularly, food and food packaging) mainly based on 2D and 3D vision data from a range of imaging modalities. It is also expected to provide useful and/or user-specified information related to the objects. In short, it should as much as possible mitigate the manual effort required by various quality control (QC) systems and operations utilised in current food industry, leading to a more flexible approach which is applicable to many different types of food products and processes.

On one hand, this technical quarterly report is a summary of the work we have done in August, September and October 2013. We integrate the corresponding three monthly technical reports in chronological order as Chapters 2-4. On the other hand, Chapter 5 raises some technical issues which emerged in our work in the past three months and also gives some suggestions to solve them. It also lists several new techniques potentially involved in the future development of the TADD system, which gives us a hint to guide our future work.

Chapter 2

TADD Technical Report for Aug 2013

2.1 Abstract

In this project report, we propose a method which can partially address the problem that we raised in the last report: semantic labelling based on image segmentation. Note that due to the lack of ground truth, we have not implemented any experimental evaluation or quantitative comparison for the proposed method. However, preliminary visual results demonstrate that our method is potentially an effective solution, or at least a promising one that we can work with towards a prototype, to the 'Trainable Vision-based Anomaly Detection and Diagnosis' (TADD) project.

We also summarise the findings revealed in the development of the reported technique, which forms a solid base for the next stage of the project and gives helpful hints for further improvement of the current method.

2.2 Review

We first review the targets of the 'Trainable Vision-based Anomaly Detection and Diagnosis' (TADD) project. The key features of TADD are shown in Fig. 2.1

In the last report, we provided feasible methods to calculate tray size and the

Software Phase 1	
Key feature	Development feature
Position of tray	What is the orientation of the tray on arrival to the machine.
Tray size	Auto detection of the tray to auto set speeds, timings of sensing and reject.
Detecting top film on a tray.	Printed, clear.
Detecting seal width on a tray.	Measures min/max width. Adjustable by simple +/- setting.
Detecting film position on a tray.	Overhang of film with tolerance. Bowed tray with shrink. Printed film position.
Detecting label on a tray.	
Detecting label position on a tray.	
Barcode detection	
Price detection	Auto character recognition with tick box for fixed or variable data.
Weight detection	Auto character recognition with tick box for fixed or variable data.
Date detection	Auto character recognition with tick box for fixed or variable data.
Detecting other text	
Product in seal detection	Adjustable by example setting. Pure visual, laser scattered light and stress pattern analysis.
Product ID for diverger/converger	

Figure 2.1: University of Lincoln tasks in Software Phase 1, as taken from the Second Level Project Plan

orientation of tray. In this report, we shall show how to achieve other features listed in Fig. 2.1 (such as seal region auto detection and recognition, price/date tick box auto detection and recognition) in an intelligent and autonomous manner. And we demonstrate our method by implementing it on the real images offered by our project partner Ishida Europe Ltd. Basically, the method still follows the workflow that we proposed in our last report as shown in Fig. 2.2.

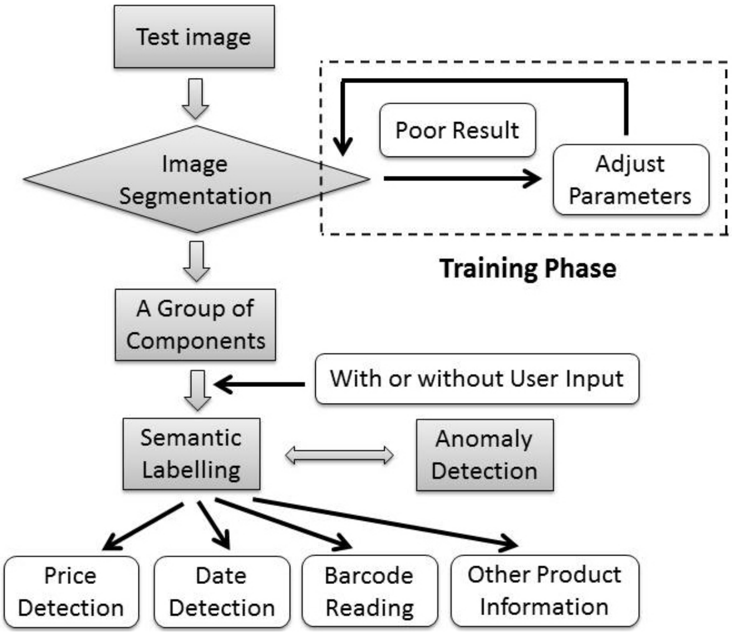

Figure 2.2: The workflow of the proposed system.

2.3 Methodology

As we mentioned in our last report, at the first stage, we selected the graph-based method [Felzenszwalb & Huttenlocher, 2004] for image segmentation, a process of partitioning a digital image into multiple components. We also revised the original method to accelerate it and demonstrated that it worked well for our specific test images through visualised results (Fig.4 in the last report). However, our further research done in August showed that it is not capable of segmenting the seal region on a tray, which means that we cannot further measure minimum/maximum width of seal as requested in Fig. 2.1. To overcome this shortcoming, we develop a method by combing a graph-based background segmentation and a superpixel-based over-segmentation. The graph-based background segmentation is actually done at a very coarse scale where image details in and around the seal region are not significantly considered. In this way, it is implemented rapidly. The superpixel-based over-segmentation cares more about local image details so that the seal region can be segmented more accurately.

In the following, we describe the proposed method to detect seal region step

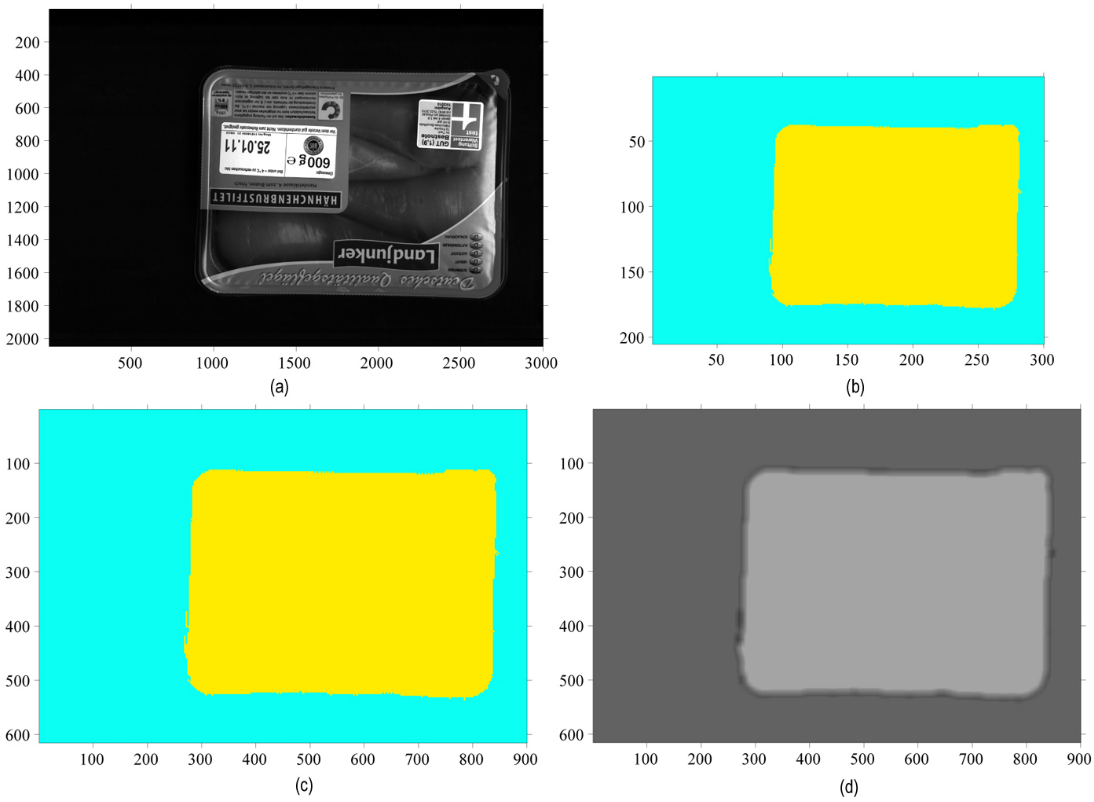

Figure 2.3: (a) The original input image (b) The binary result of background segmentation (c) The result of rescaling the background segmentation (d) Average filtering result of the rescaled background segmentation

by step.

1. First, we coarsely partition an input image into foreground and background using the graph-based segmentation method [Felzenszwalb & Huttenlocher, 2004] where the input image is downscaled by a factor of 0.1 before running the graph-based segmentation and rescaled to a larger size S after the segmentation. This process is illustrated in Fig. 2.3. Please note that in all image resizing/rescaling operations involved in this work, we use nearest-neighbour interpolation.

2. We then introduce a 2D average filter. The result of applying such a filter to the binary background segmentation is an image with a blurred boundary

as shown in Fig. 2.3 (d).

3. We rescale the original input image to size S and implement the SLIC algorithm [Achanta *et al.*, 2012] for producing superpixels on the rescaled image. The starting size of the superpixels should be set at roughly the same magnitude of the seal width. The result of superpixel over-segmentation is shown in Fig. 2.4 (a).

4. We detect the seal region by classifying the superpixels across the blurred boundary.

5. We finally rescale the seal segmentation to the size of the original input image as shown in Fig. 2.4 (b).

As we demonstrated in the last report, semantically meaningful tick boxes can be directly detected by graph-based method. In the work reported here, we take a further step towards a complete system for semantic segmentation and recognition. In order to detect and recognise the date/price tick box, we analyse the intensity information of each component for an automated and intelligent understanding of what it is. For example, the mean intensity value of a tick box should be much higher than that of the background region. Thus finally, we integrate segmentation with recognition. And the result of our method is shown in Fig. 2.5.

We have implemented the proposed method on 25 images offered by Ishida Europe Ltd. On the average, the time for processing one image is roughly 11 seconds.

2.4 Future work

Here we raise several issues for future work.

Firstly, future development can potentially focus on the acceleration of the method. Apart from software (i.e., converting the current mixture of MATLAB and C++ codes to C++ codes only) or hardware ways (i.e., using GPU and CUDA-based programming), more interestingly, one algorithmic way is to reduce the image regions that we input to the SLIC algorithm. On one hand, in our

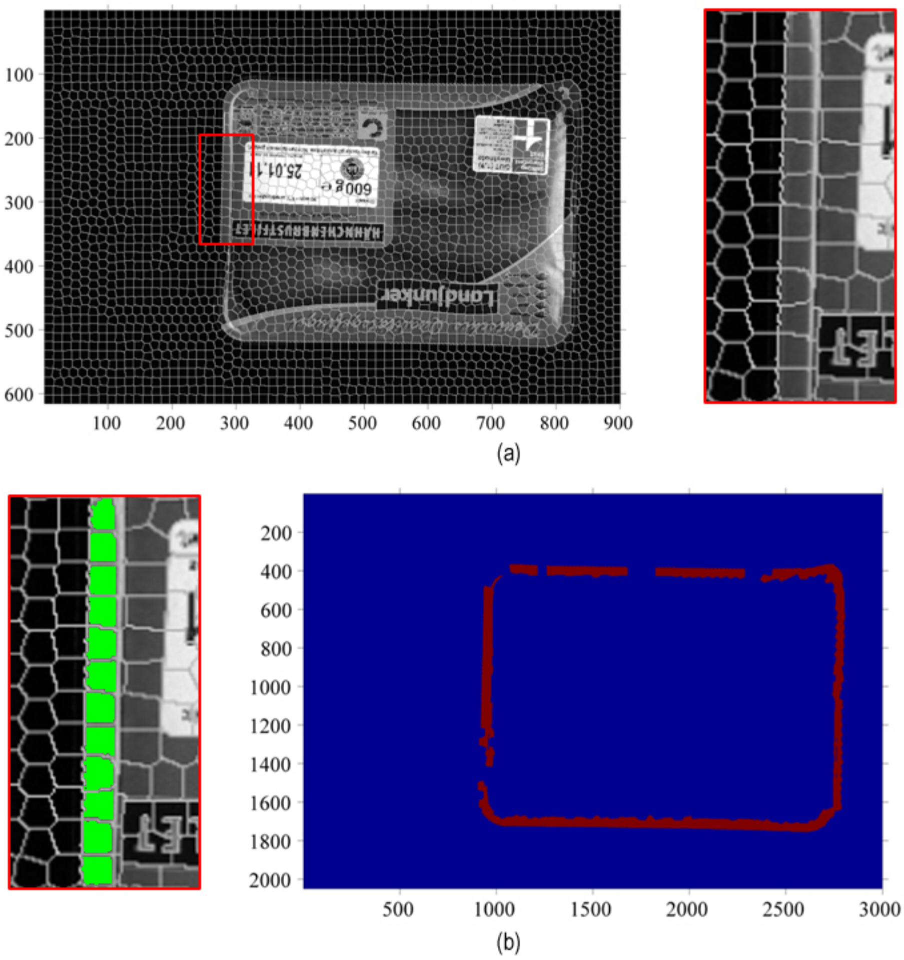

Figure 2.4: Seal region detection via superpixel classification. (a) Superpixel over-segmentation (b) Seal detection

experiments we found that SLIC cannot be implemented rapidly even through C++ on a CPU when the input image is of high resolution and it typically costs more than half of the total execution time. Note that images of high resolution are always desired from a quality control point of view and sometimes even necessary due to the requirement of character recognition (e.g., barcode detection, price detection, date detection, etc). On the other hand, we notice that in the tray images, the superpixel over-segmenation of some regions is actually not necessary. For instance, the over-segmentation in the background region is always pointless in any case. Also, if we just need SLIC for segmenting seal region, obviously a

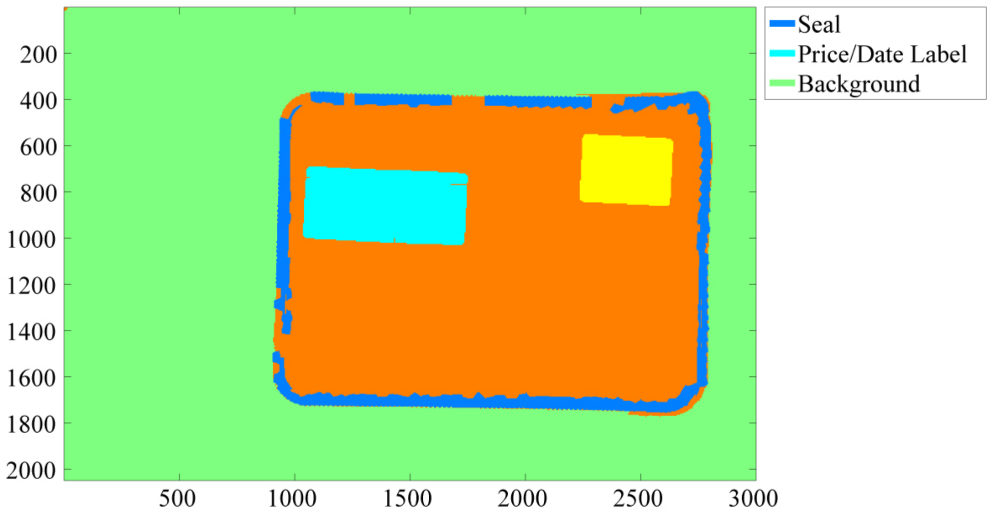

Figure 2.5: The final output of the proposed method where the input is the image shown in Fig. 2.3 (a). The entire processing is fully automatic.

large image region inside the tray can be ignored. In short, we may first define a (potentially as small as possible) region of interest (ROI) for the SLIC algorithm and then merely implement it in the ROI. Nonetheless, the detection of ROI need to be automatic and ideally it should consist of a collection of rectangular image regions because the initialisation of SLIC requires a regular grid.

Secondly, to more accurately detect and recognise each component in an image, training is definitely needed. Training could be very important for the fine segmentation of a high-resolution image. Once the user selects a specific collection of pixels (e.g., superpixels) and labels them, the system should be able to vectorise the intrinsic features of the selected superpixels and improve current segmentation/classification based on them. For example, the user can select a collection of pixels in the seal region and compute their features such as intensity, colour, gradient, etc. If some pixels automatically labelled as 'seal' by the system have very inconsistent features, the segmentation system (essentially a classifier) can be trained to relabel such pixels. Besides such kind of online interactive training, offline training is also very useful for parameter settings. This will significantly ease the manual burden for fine tuning the controlling parameters of the algorithm and reliably produce good segmentation results.

Thirdly, the ground truth data are always required for both scientific evaluation and the aforementioned offline training. Therefore, in the future, we also plan to construct some ground truth data by manually labelling the tray images.

Finally, we have tested the seal detection method on colour images produced by a low-cost webcam and observed improved results compared with using corresponding grey images as the input. Hence, future work will contain the exploration of the utility of colour images.

Chapter 3

TADD Technical Report for Sep 2013

3.1 Abstract

We report our improvements over the original TADD system which was initially designed for detecting defects and blemishes on potatoes. Also, such improvements are analysed through small-scale experiments using the image acquisition equipment and lighting settings of the original TADD system. In general, the technical improvements are based on our previous work done in July and August. However, most of our previous work were implemented in MATLAB. In September, we have reimplemented them fully in C++ for fast implementation and integrated them with the original TADD system.

3.2 Introduction

Technically, the TADD system is a prototype low-cost machine vision system which generally lays emphasis on real-time implementation. To meet the industrial standard of software development, it incorporates an intuitive graphical user interface (GUI). These two points make it a desirable platform to be augmented with the techniques that we specifically develop for the vision-based anomaly detection on food trays (e.g. graph-based background segmentation, intelligent

Figure 3.1: The backgrounds of tray images are often non-black and inconsistent.

detection for tray position and orientation and semantic labelling, etc[1]).

In September, based on our previous work, we make some significant improvements on TADD. They are mainly twofold:

1. We replace the original thresholding-based background segmentation with the newly developed graph-based segmentation and it also achieves the real-time implementation. The huge advantages of graph-based segmentation will be discussed later in this report.

2. To meet the requirements of the project, we integrate fast contour and bounding box detection of the tray with the TADD system to visualise its position and the orientation.

3.3 Review

The original TADD system consists of both hardware and software components. Its off-the-shell hardware comprises of a low-cost webcam of 60Hz, a standard desktop PC with GPU and a light chamber with 2 LED bulbs inside. The entire software system including the image capture unit, the background segmentation unit, the superpixel-based over-segmentation unit, the feature-generation unit,

[1]Please refer to our previous project reports for details.

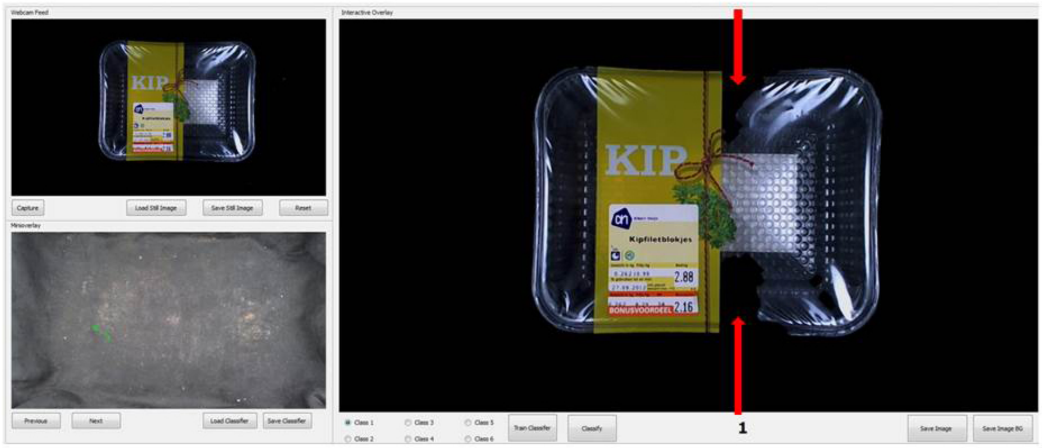

Figure 3.2: Some regions of the tray are incorrectly regarded as background region by the original TADD system. And thus users cannot select superpixels within those regions for the following classifier training and classification. Note that according to the system design, users should be allowed to select superpixels in the region of interest (ROI) and be prohibited from doing so in the background region.

the Adaboost-based classifier training unit, the classification unit and the quality analysis unit, is based on such hardware, in particular, the settings inside the light chamber. For instance, to achieve an efficient and accurate background segmentation, the original TADD actually relies on a black felt which is used as the background and greatly reduces the complexity of background. However, in our project, the background colour is not always black and consistent as shown in Fig. 3.1.

The original TADD system employs simple thresholding for background segmentation, which is obviously based on the assumption that the object of interest has higher intensity values than the background. Perhaps this is true for potatoes. However, in our project, such an assumption will lead to errors illustrated in Fig. 3.2. This error could frequently occur since it is usual that the tray is not fully filled and thus transparent, which means some regions within the tray might have similar intensity to the background. In addition, Fig. 3.3 shows that the original TADD system cannot handle non-black background.

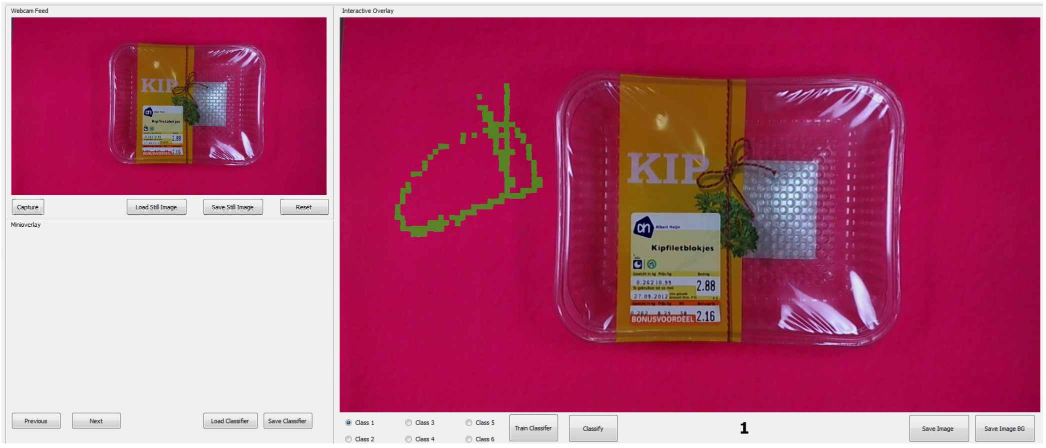

Figure 3.3: The original TADD system cannot handle non-black background.

3.4 Our method

Aiming at the aforementioned limitations of the original TADD system, in our system, we incorporate the graph-based method for background segmentation. The graph-based method treats the whole image as a graph consisting of nodes and edges. Therefore, it does not only consider the independent properties of each node (such as intensity value), but also utilises the connectivity information between nodes. In this way, the algorithm can recognise that a black pixel within the ROI (i.e. the tray region) is different from a black pixel in the background. The improvement delivered by the graph-based background segmentation is significant. To visualise such improvement, and to more clearly answer the key issues (position of tray and tray size) raised in the project plan, we also developed some code to visualise the contour and the bounding box of the tray. Figs. 3.4 and 3.5 show the results of our background segmentation as comparisons to the background segmentation of the original TADD.

The new TADD system is actually more powerful as demonstrated in Figs. 3.6, 3.7 and 3.8. It is worth noting that all of these implementations are fully automatic and work in real-time.

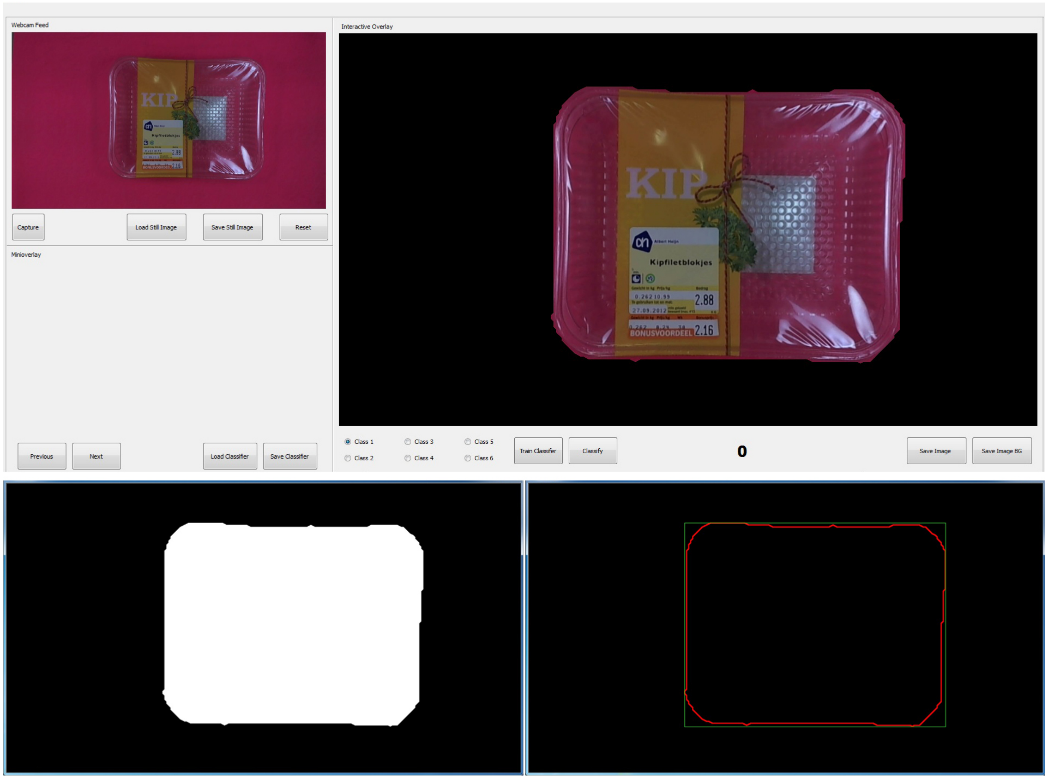

Figure 3.4: Background segmentation result using our system. Please compare it with Fig. 3.3.

3.5 Future work

Although with the graph-based background segmentation method, the new version of TADD is more powerful, there is still a large room to improve it, making the system more accurate and reliable. For example, we currently use the default parameter settings for the graph-based background segmentation. Further work will focus on the development of a training-based parameter selection mechanism (two key parameters to be trained). Also, in our experiments, we found that the reflection caused by the LED light sometimes forms a significant interference and potentially makes the background segmentation a difficult task. Considering that the current light chamber is specifically designed for anomaly detection on potatoes, we are waiting for the new sensor (with the capability of providing colour and range information) believed to be more suitable for our project.

To catch up with the task of barcode detection mentioned in the project plan,

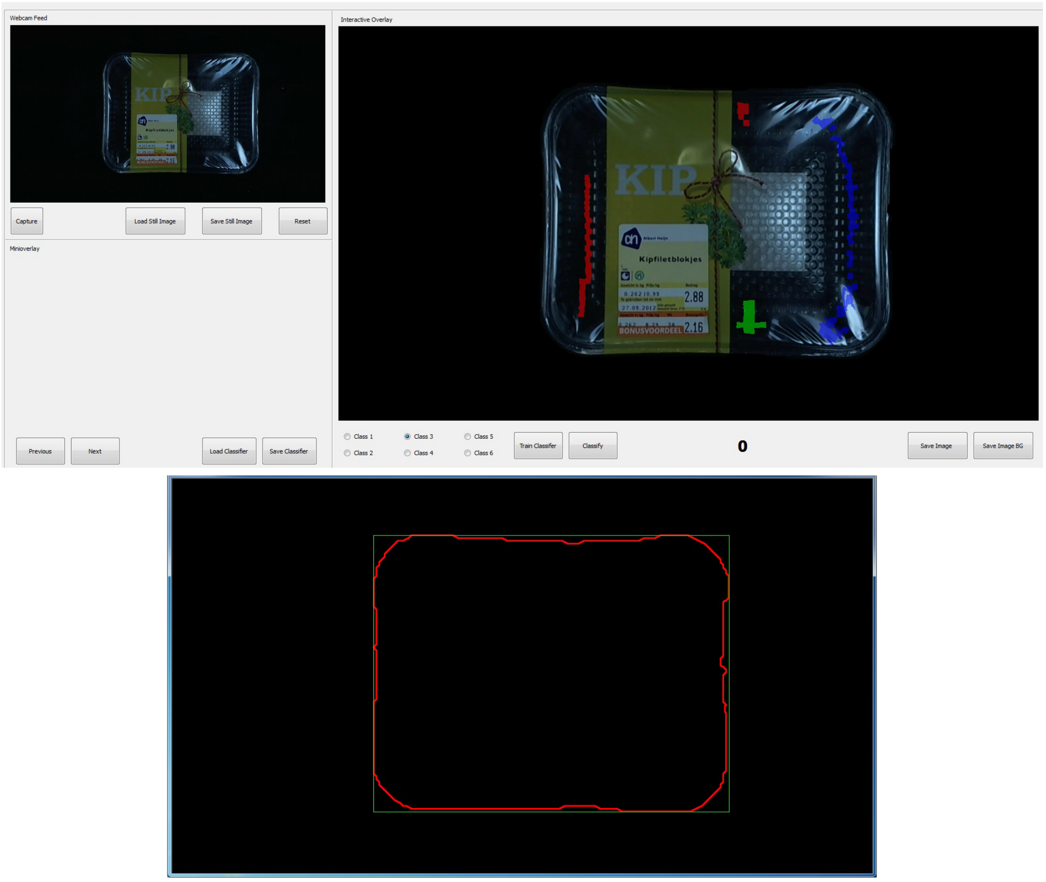

Figure 3.5: Background segmentation result using our system. Please compare it with Fig. 3.2.

we also review some papers in the related field. More accurately, it should be called 1-D barcode localisation in complex backgrounds. It typically requires various region-based image analysis algorithms. It is actually a research topic very different from the well-known barcode reader techniques and quite new in computer vision and computational intelligence. We can see that lots of researchers are focusing on it due to the huge commercial interest behind it. Note that currently, the prevalent barcode readers are devices with laser-scanners which require that laser-scanners must be close enough to the barcode region (so that the scanned non-barcode region is small). With the development of artificial intelligence and various personal mobile devices, people are paying more and more attention to computer vision, pattern recognition and human-computer interac-

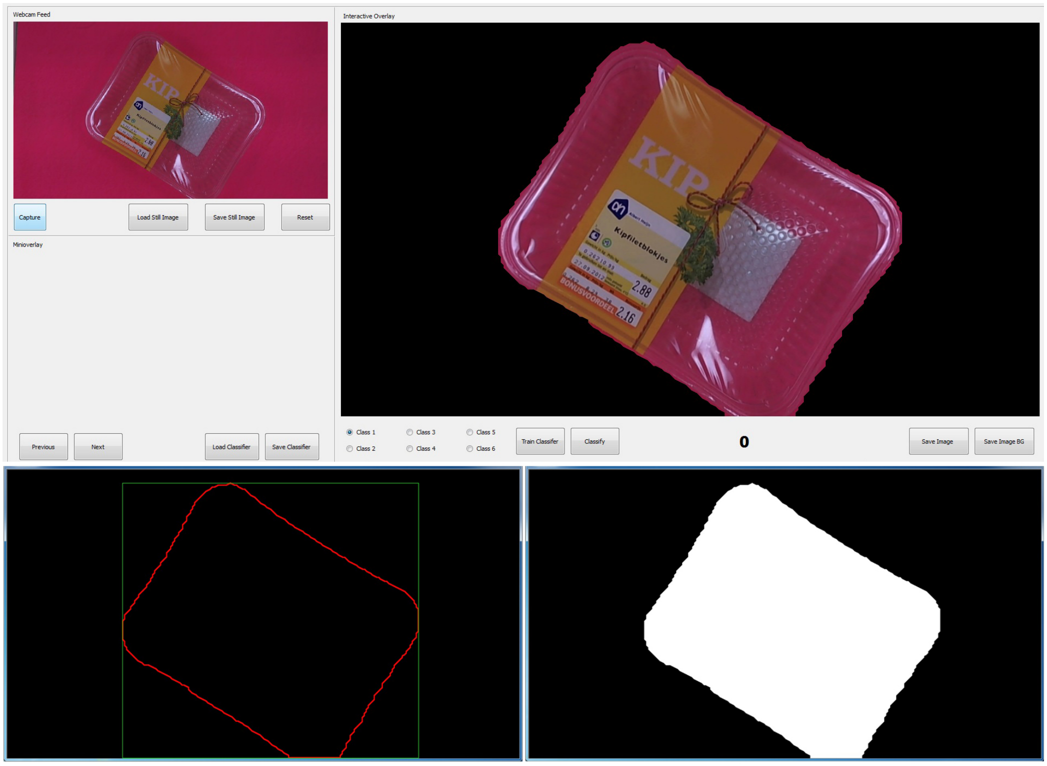

Figure 3.6: Detection of tray position and size with a non-black background.

tion. Recognising barcode by understanding image context is a new tendency. [Fang & Xie, 2010] is a short paper where the authors proposed an algorithm for barcode localisation in the spatial domain of an image. Hence, barcode localisation in complex backgrounds will be definitely within our future work plan.

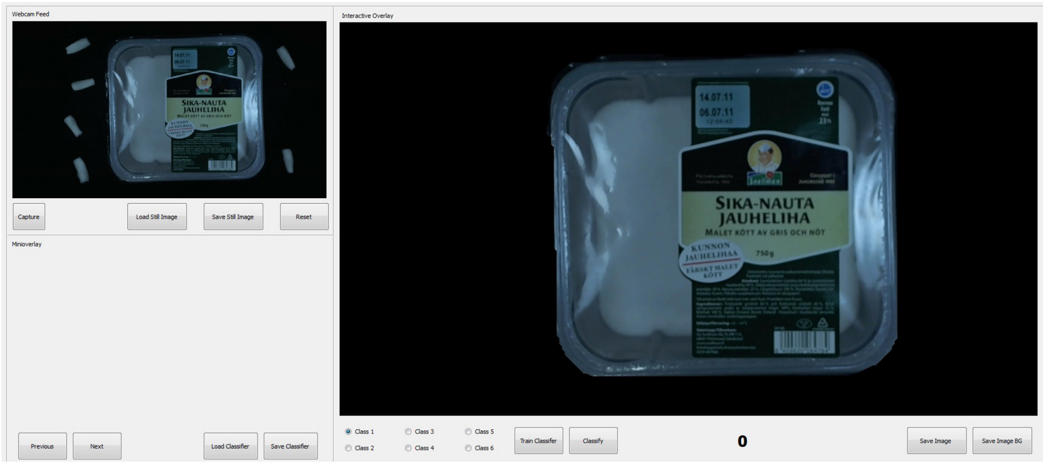

Figure 3.7: Background segmentation on an image with low brightness and some objects of interference.

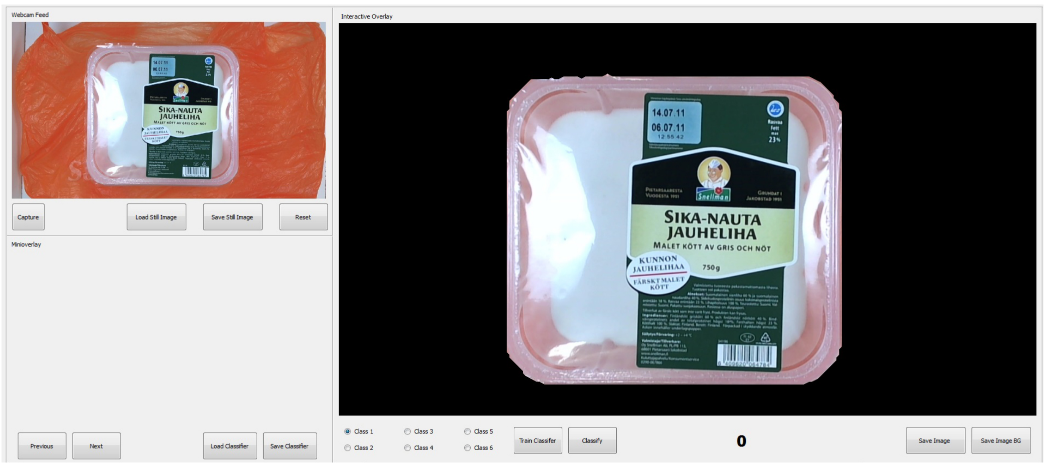

Figure 3.8: Background segmentation on an image with non-black and inconsistent background

Chapter 4

TADD Technical Report for Oct 2013

4.1 Abstract

This report is twofold. One the one hand, we summarise our new improvement on the current TADD system, which is based on the well-known Scale Invariant Feature Transform (SIFT) algorithm. On the other hand, we propose an efficient method for detecting barcode region within a food tray from a complicated background and explain the related technical details.

4.2 Review

In our last report, we show an improved TADD system which can automatically, intelligently and efficiently partition a food tray image into background and foreground regions, even if the background is inconsistent, non-black or with noisy objects. After such a background segmentation, the system visualises the position and the orientation of the tray by detecting the contour and the bounding box of the tray.

The segmented tray region is actually the region of interest (ROI) where we can (i) more efficiently implement other demanding operations within it (e.g., tickbox detection, label detection and seal detection, etc) since we just need to

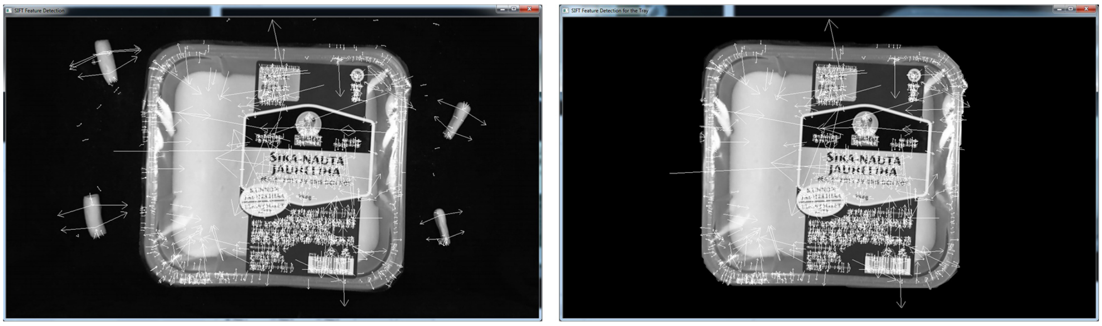

Figure 4.1: SIFT feature detection. Left: without background segmentation; Right: with graph-based background segmentation

handle a smaller number of pixels by ignoring the pixels in the background region and (ii) more reliably implement these operations because it is highly possible that there are some noisy pixels in the background region which have an effect on the implementation of the operations in the tray region.

Thus in this work, we apply SIFT algorithm [Lowe, 2004] on the input images and integrate the codes with the current TADD system. The detected features can be potentially used for various tasks in the future development of the TADD system.

4.3 SIFT feature detection

As one of the most famous and widely-used computer vision algorithms, SIFT algorithm transforms image data into scale-invariant coordinates relative to local features. Typically, it can generate a large number of features that densely cover the image over the full range of scales and locations. Each feature has three visualised properties: scale, orientation and location. In fact, typically, each feature corresponds to a vector of high dimensionality (which represents the image gradients within a local region of image where the size of the local region is defined by a Gaussian) and thus the SIFT features are very distinctive and can be matched reliably. Experiments implemented in [Lowe, 2004] demonstrate that SIFT features are invariant to image scaling and rotation, and partially invariant to change in illumination and 3D camera viewpoint.

Fig. 4.1 shows two images of SIFT features produced by the improved TADD system. In Fig. 4.1, we use an arrow to represent a detected SIFT feature. The starting point of the arrow denotes the location of the SIFT feature; the size of the arrow denotes the scale of the feature; and the orientation of the arrow is the orientation of the feature. It can be observed that the background segmentation effectively removes the features detected on the noisy objects within the background region. This is particularly important for the following object recognition (tickbox detection, label detection, etc) since these noisy objects could have similar features to those within the ROI and they can thus lead to unreliable matching.

4.4 Barcode region detection

Barcode region detection, or more accurately, automatic barcode region detection in complex background, is a technique very different from the current prevalent barcode reading techniques. Barcode reading techniques usually rely on devices with laser-scanners and assumpts that the barcode region are dominant over the entire image. However, this is not the case in our project. As shown in the top row images of Fig. 4.2, the barcode regions are quite small in size compared to the entire image. To find the barcode region in complex image background efficiently and reliably, we propose a new algorithm. The basic idea is to capture the difference between the horizontal gradient and the vertical gradient of the image. Then the barcode region should correspond to the largest differences due to the specific shape and intensity distribution within a barcode region. The details of the algorithm is shown in Algorithm 1.

According to Algorithm 1, the proposed method for barcode region detection is quite fast. Also, as demonstrated in Fig. 4.2 (b), this method has some degree of robustness to rotation. If the rotation angle is quite large (although it is extremely rare in our project), we perhaps need to first rotate the tray according to its orientation information captured by the background segmentation.

One significant limitation of our current method is that the boundary of the barcode region is typically not well defined. One solution to this problem is to give a tolerance to the detected region (e.g., ± 50 horizontally and vertically) and

Algorithm 1: Barcode region detection in complex image background

Data: A 2D image I

Result: A saliency map I

begin

  if I *is an RGB image* **then**

    convert I into an intensity image;

  Compute the horizontal and the vertical gradients of I, writen as I_x and I_y;

  Measure the difference $D = abs(abs(I_x) - abs(I_y))$ where abs denotes the computation of taking the absolute value of each element in a matrix;

  Construct an average filter kernel H with the size of 60×60;

  Do convolution using H and the difference map D: $C = H \otimes D$;

then to calculate the rectangular bounding box of the region. The image content within the bounding box is in fact a new image where the barcode region should be dominant and thus general barcode readers can work well with it.

Certainly there are some other more advanced techniques which can more accurately detect the boundary of the barcode region. For instance , [Fang & Xie, 2010] achieved this through region-based image analysis. Nonetheless, our method is more efficient. Typically it can detect the barcode region in a complex 3000×2048 image within 1.5 seconds on a standard PC (Duo core, 3GHz CPU, 2GB RAM).

4.5 Future work

Since the current TADD system can correctly detect SIFT features in the tray region, tasks like tick box detection can be done by matching the detected SIFT features with those saved as template features. Because SIFT features are invariant to image rotation, this strategy should be robust as long as the template features can be extracted. Once the feature matching is implemented, the system can estimate whether the positions of the tickboxes are correct or acceptable by checking the locations of the features within those tickboxes.

Another way to do semantic object recognition is to decompose the tasks.

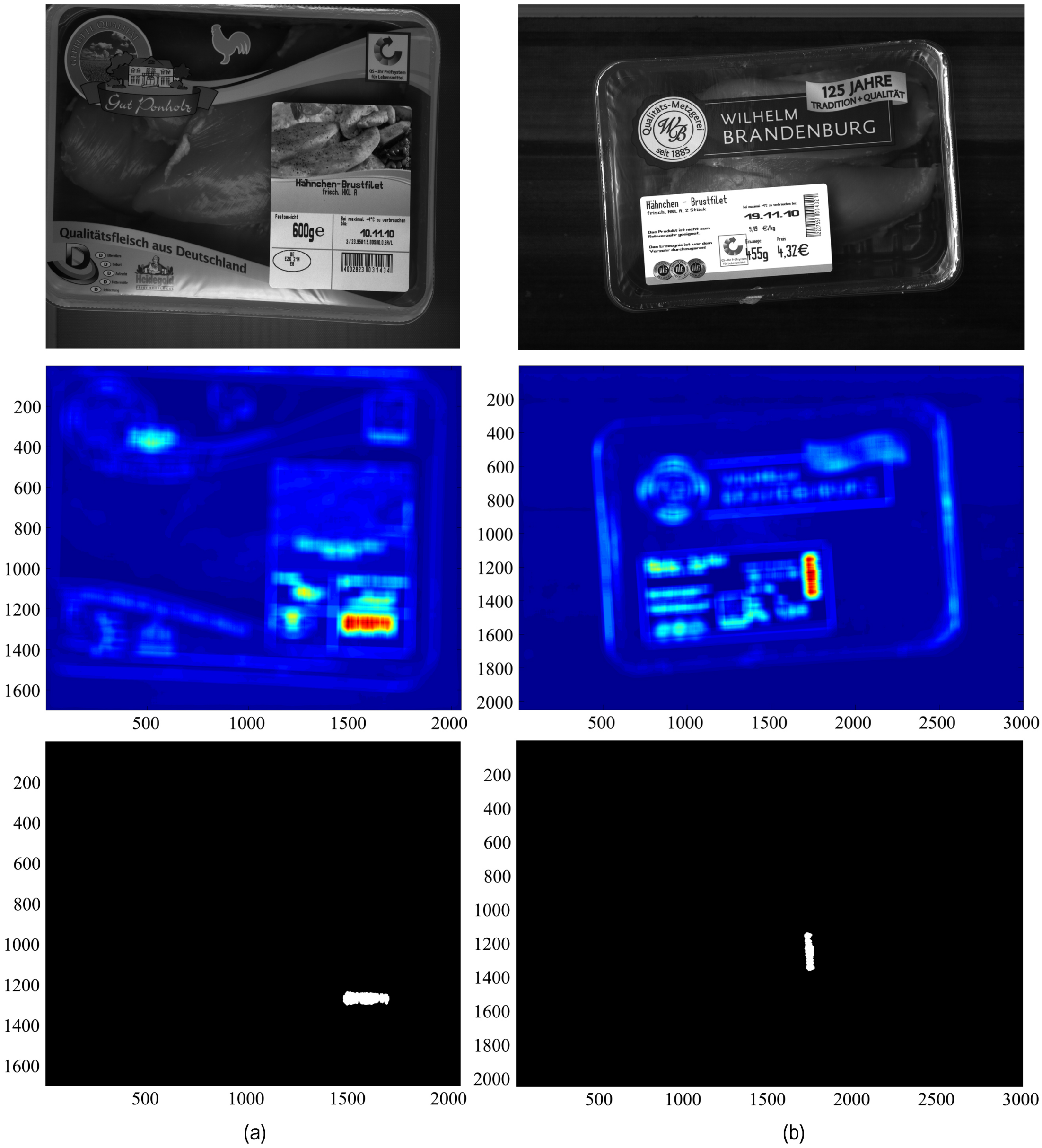

Figure 4.2: Results of our method for barcode region detection. (a) A horizontal barcode (b) A vertical barcode. Top row: Original input images; Middle row: Colour maps (C in Algorithm 1) produced by our method where warm colour represents strong response to potential barcode region; Bottom row: Binarisation of the colour maps which clearly locate the barcode regions (The binarisation threshold is set as $T = 0.7 \times C_{max}$ where C_{max} is the maximum value of C).

At each step, the system just focuses on one specific tickbox. For instance, the system can first read in the image with a blank tray (with patterns but without tickboxes) and then read in a tray with only one tickbox. Next, by comparing the difference of SIFT features detected in the two trays, the system can know what the tickbox is. The potential drawback for the latter method is its low speed. It needs to perform several rounds of SIFT algorithms since each newly added tickbox or label will change the content of the tray surface and consequently change the result of SIFT detection. Considering that SIFT does not have a real time performance, the entire processing time might be too long. Therefore, perhaps the next step is to find and test some other feature detection algorithms (such as SURF: speeded-up robust features [Bay *et al.*, 2006]) faster than SIFT using the specific tray images taken by our TADD system.

Another interesting work is to improve our current barcode region detection algorithm. In particular, we hope to detect the boundary of the barcode region more accurately at a low computational cost.

Chapter 5

Conclusions

In this chapter, we draw some conclusions and raise several technical issues based on our work in the past three months.

Considering that one important criterion to evaluate the TADD system is its running time, we concentrate on the time performance for each component of the current TADD system. In summary, it has been demonstrated that the graph-based background segmentation is powerful and reliable in most cases. However, it is not so fast as the original thresholding-based method and usually costs 1–2 seconds, depending on the resolutions of input images. Therefore, one potential solution is to keep both methods optional (or switchable and we can put the simple thresholding-based method as default) because it is not realistic to reduce the running time of the technically complicated graph-based background segmentation to the level of a simple thresholding-based method. In that case, our end user can opt for the one which can meet their specific requirements in practice.

SIFT feature detection that we currently employe for the TADD system is computationally expensive. In our implementation, SIFT typically took 3–4 seconds to extract hundreds of features through one input image. This seems not acceptable for TADD because we pursue a system which can get all things done in a real-time or near real-time rate. For this reason, we have been doing some research on other feature detection techniques. The codes of another feature detector, SURF (speeded-up robust features) [Bay *et al.*, 2006] has been integrated with the current TADD system. Preliminary tests show that it can achieve a real-time performance (the detection was completed almost immediately after we clicked the corresponding button in the interface of TADD) and more experimental results on SURF will be released in the technical report for November 2013.

The input of the current TADD system is purely 2D. We plan to move to 3D in the next stage of the development since one of our conclusions for the previous work is that 3D information is definitely needed for TADD. On one hand, 3D input information can simply solve some problems that 2D information cannot solve. For instance, for some food products we want to measure the volume of the food assigned in a tray, which cannot be handled without 3D information. 3D information is obviously helpful for seal detection and analysing anomaly in the seal region. Also, some anomalies, such as the undesired sprouts on a potato, can be more easily detected by using 3D information. On the other hand, 3D information can be used to improve the accuracy of the computation based on 2D information in the TADD system. For example, the current TADD system finally output the percentage of each object class by computing their areas respectively. Note that the areas computed based only on a 2D image is not accurate due to the effect of projection. For a potato, since it is usually not flat, such a computation for surface areas could cause significant errors. Because every 2D image is actually a 2D projection of a 3D object in the real world, it is necessary to take 3D information into account in most cases.

Therefore, future work should figure out several issues listed below:

(1) What type(s) of 3D input data do we require? Point clouds? Depth maps/range images? Meshes? etc...

This is quite important since it actually shapes the mechanism of the system. For instance, there are two types of range images: with or without colour. If we use the one without colour, we also need to develop some algorithms to do registration or texture mapping.

(2) How can we integrate 3D with 2D?

This is potentially a large chunk of workload in the next stage.

(3) Now we have more data to deal with. How can we still make the system run fast?

3D data is typically much larger than 2D data. The range image (with the resolution of 3000×4000) of a potato produced by Cognex last week in our office in Lincoln is 68.6MB, much larger than a 2D image with the same resolution.

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P. & Susstrunk, S. (2012). Slic superpixels compared to state-of-the-art superpixel methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **34**, 2274–2281. 6

Bay, H., Tuytelaars, T. & Van Gool, L. (2006). Surf: Speeded up robust features. In *Computer Vision–ECCV 2006*, 404–417, Springer. 23, 24

Fang, L. & Xie, C. (2010). 1-d barcode localization in complex background. In *Computational Intelligence and Software Engineering (CiSE), 2010 International Conference on*, 1–3, IEEE. 16, 21

Felzenszwalb, P.F. & Huttenlocher, D.P. (2004). Efficient graph-based image segmentation. *International Journal of Computer Vision*, **59**, 167–181. 4, 5

Lowe, D. (2004). Distinctive image features from scale-invar-iant keypoints. *International Journal of Computer Vision*, **60**, 91–110. 19