TSB-funded Project 'TADD' - Trainable vision-based anomaly detection and diagnosis Technical Report for 2nd Quarter

(*1 July 2013 — 31 July 2013*)

Ran Song and Tom Duckett

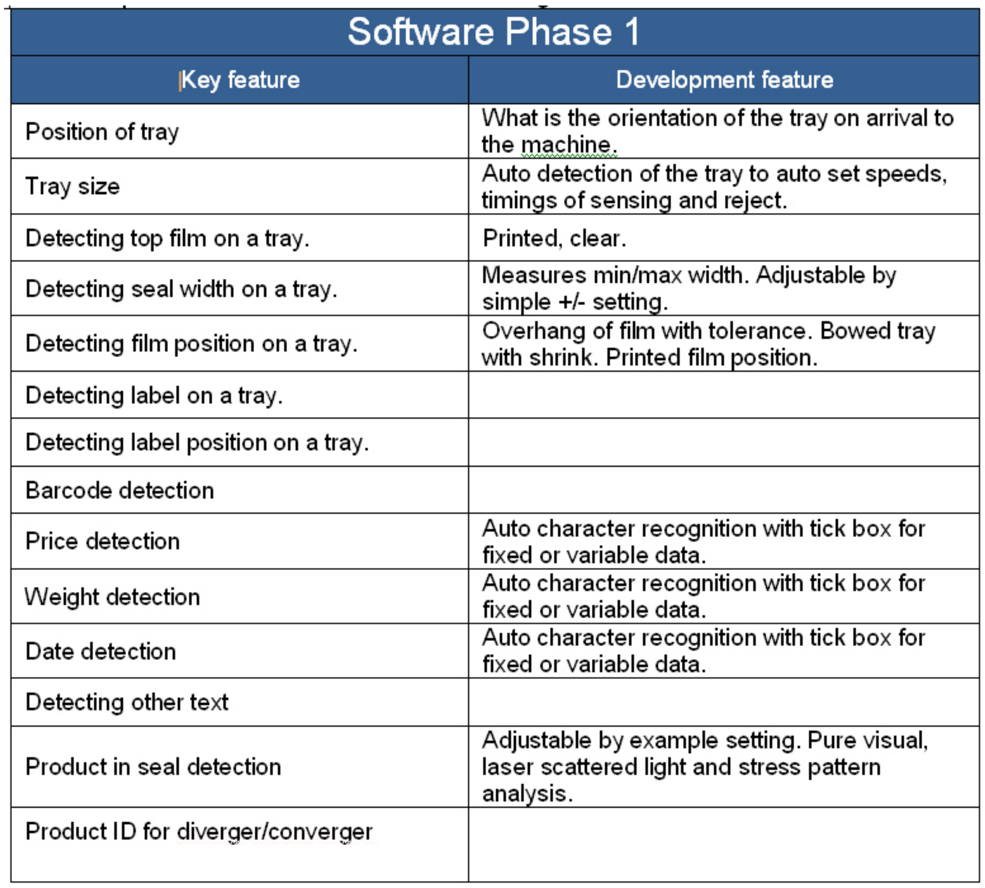
Agri-Food Technology Research Group, Lincoln Centre for Autonomous Systems Research, School of Computer Science, University of Lincoln, UK

Abstract

This report documents the preliminary findings, analysis, experiments and results for the 'Trainable Vision-based Anomaly Detection and Diagnosis' (TADD) project completed in the month of July 2013. The report gives an overview of a potential approach to meet the project requirements, focussing in particular on the problem of *image segmentation* in order to address many of the key features listed in Phase 1 of the Second Level Project Plan. This is followed by conclusions and future work for the next project quarter.

1. Problem statement

The 'Trainable Vision-based Anomaly Detection and Diagnosis' (TADD) project aims at developing a system, delivered as software package, which can automatically and efficiently detect and recognise anomalies on the considered objects (particularly, food and food packaging) based on 2D or 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.



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

The development of such a system is closely related to several major topics in the research fields of computer vision and machine learning: 2D/3D image segmentation, data classification and object recognition. However, due to its specific application including specific input data and specific system requirements, as we found through our preliminary work, current mainstream computer vision approaches to the related topics usually proposed for handling general input data and achieving broad utility do not work well on our test images. Another problem is the speed of implementation. Most of the state-of-art image segmentation and classification algorithms cannot be implemented at real time speeds when processing high-resolution images.

This report focusses mainly of the problem of *image segmentation*, which is the process of partitioning a digital image into multiple components. This will be an essential component of the overall TADD system because it is needed as a pre-processing step for implementing many of the key features listed in Fig. 1. In particular, image segmentation will be required as a pre-requisite for solving the following tasks:

- position of tray,
- tray size,
- detecting seal width on a tray,
- detecting label on a tray,
- detecting label position on a tray,
- detection of barcode, price, weight, date and other text,
- etc.

2. Current Ishida vision system

The current vision-based system embedded in the QC (quality control) equipment offered by the project partner Ishida Europe is based on Sherlock[1]. A major shortcoming of the system is that, as mentioned in its tutorial, it always requires the user to manually create a region of interest (ROI), typically a rectangle at the first place, as shown in the left image of Fig. 2. In practice, due to the variety of the objects (e.g., various food trays, potatos, etc.), and/or the various orientations of these objects, the shapes of ROIs are potentially varying. And thus it could be difficult and time-consuming for the end users who are usually not experts on computer vision to manually define the ROIs. More importantly, there might be a couple of ROIs in one image (e.g., text regions, labels on a food tray, etc.). In this case, it is very difficult to locate each ROI manually.

After defining ROIs, Sherlock software allows the user to assign algorithm operators to ROIs. The right image of Fig. 2 shows an example of thresholding. Note that here users also make a great effort: they tell which ROI

[1]http://www.teledynedalsa.com/imaging/products/software/sherlock

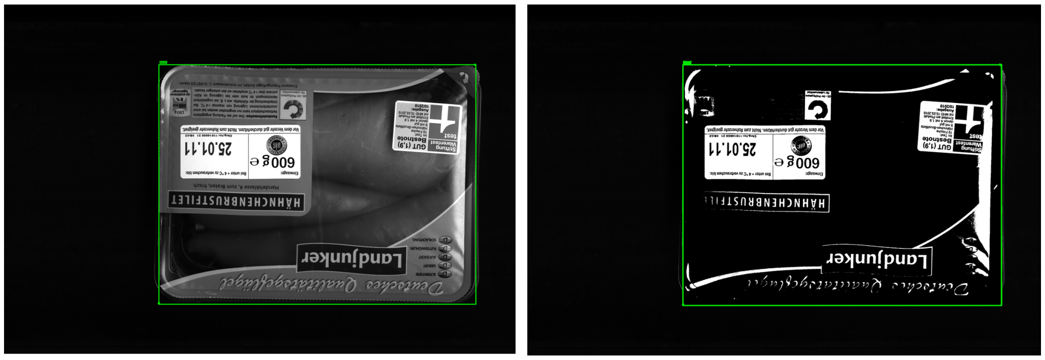


Figure 2: Left: greyscale input image to Sherlock system; Right: result of thresholding operation

should be assigned what computer vision algorithm. Such a clue is significantly important for the system but it would not be easy for a non-expert to deliver. In addition, different algorithms usually involve various parameters which might have significant impact on the outcomes. For end-users from food industry with little know-how to computer vision, it is not just time consuming, but in fact extremely difficult to use Sherlock system to deliver good QC.

3. The proposed system

The proposed system is motivated by the shortcomings of Sherlock and specifically designed for the key features of development in TADD. Fig. 3 illustrates its workflow. Basically, we replace the manual effort for ROI selection and operation assignment required by the Sherlock system with more automatic and intelligent computer vision techniques.

3.1. Image segmentation

Image segmentation is the process of partitioning a digital image into multiple components. We have tested various algorithms published in the past couple of years and found that the graph-based segmentation method [1] worked well. One major problem of this method is its computational speed although it was claimed to be efficient. For the test image shown in Fig. 4(a) with a resolution as high as 3000×2048, it took roughly 2 minutes to produce the segmentation result shown in (b). The original code of the graph-based segmentation method has thus been modified by adding code for

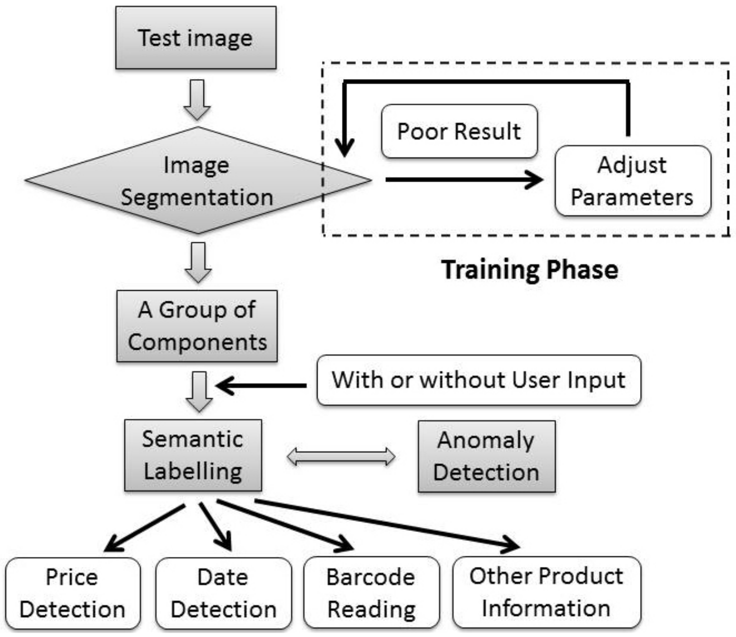
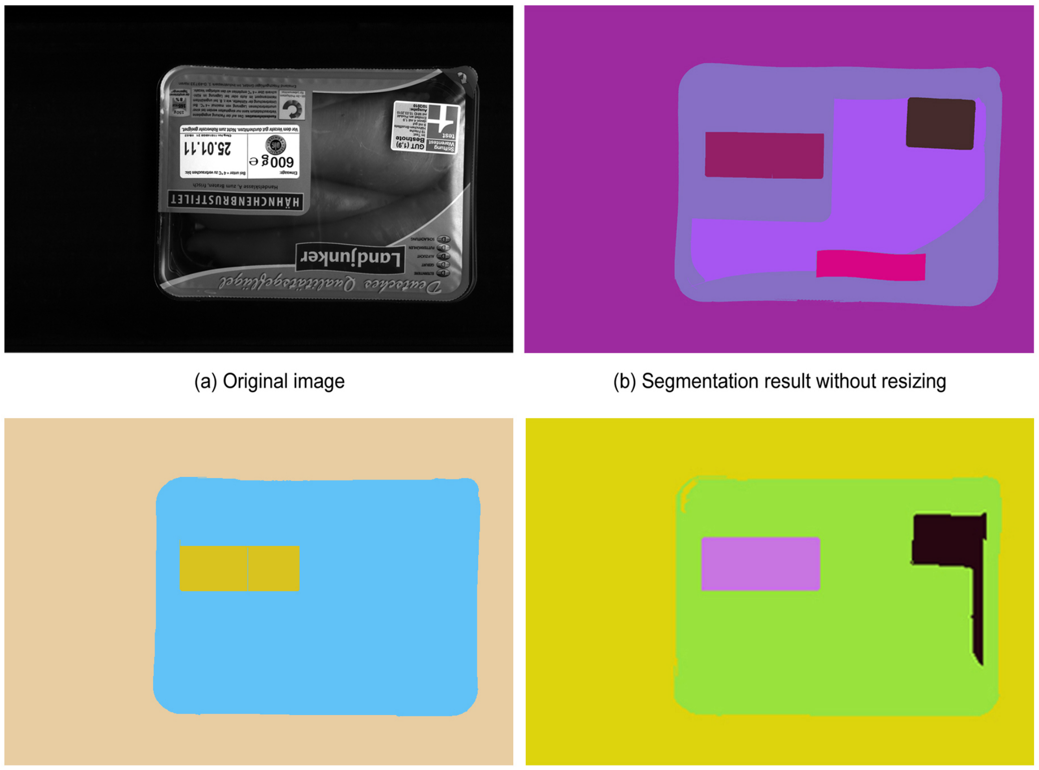


Figure 3: The workflow of the proposed system.

image resizing (and also code for reading and transferring BMP images since the original program can only deal with PPM images) using OpenCV. As shown in Fig. 4(c) and (d), the segmentation is still good. But the algorithm has been significantly speeded up. It took 8 seconds for the case of (c) and 0.8 seconds for the case of (d).

Fig. 4 demonstrates that (1) at least background region of our test images can be reliably segmented using automatic image segmentation algorithms with proper parameter settings (2) such an algorithm can be significantly speeded up with very minor effect on background segmentation by scaling down the input images. This is because the computational complexity of the segmentation algorithm is not linear in the number of pixels. In the experiments, we also scaled down the image by a factor of 0.04 and found the background region can still be reliably segmented. The background segmentation already gives us some useful information. For instance, the tray size can be calculated by counting the number of pixels in the foreground region. And the orientation of the tray can be calculated by first finding the two pixels (x_1, y_1) and (x_2, y_2) with the minimum and the maximum y-coordinates over all foreground pixels and then computing $\alpha = |\frac{y_2 - y_1}{x2 - x1}|$. If α is close to


(a) Original image

(b) Segmentation result without resizing

(c) Segmentation result after resizing by 0.3 * width (height)

(d) Segmentation result after resizing by 0.1 * width (height)

Figure 4: Segmentation results using different resizing settings. Note that the number of components can be adjusted by using different parameters (namely σ, k, and min as mentioned in [1]).

the ratio between the height and the width of the image (in this case it is 2048/3000), it means that the tray is in a good position on arrival to the machine.

Another problem of using such a generic approach to image segmentation is that it does not take into account application-specific information which could be provided by a non-expert human operator, including the number of desired categories for the segments (background, tray, label, food, etc.) and some examples of each category (e.g. a few areas in the image labelled by the human operator as examples of the background, tray, label, food, etc.). By including this information, a segmentation algorithm could potentially be made much faster and more robust.

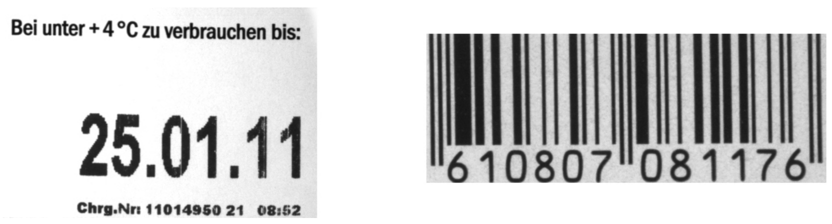

Bei unter + 4 °C zu verbrauchen bis:
25.01.11
Chrg.Nr: 11014950 21 08:52
610807 081176

Figure 5: Left: A label; Right: A barcode

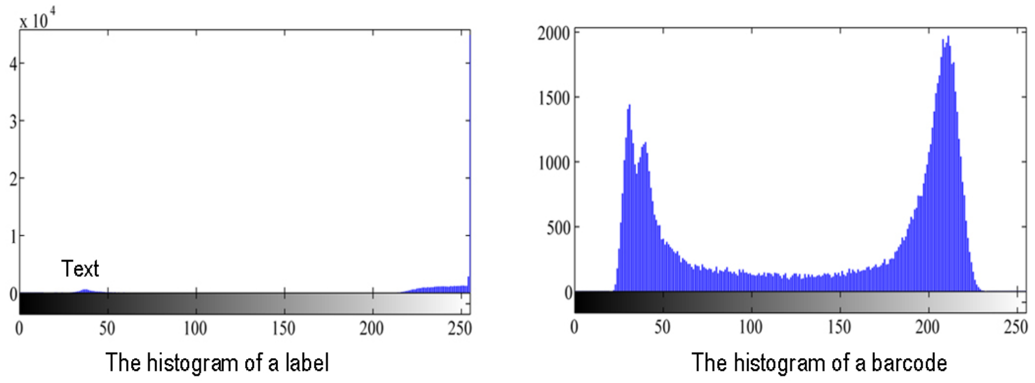


Figure 6: The histograms of a label and a barcode

3.2. Semantic labelling

Another related image processing operation to be developed in the project is semantic labelling. Its target is to tell the computer what an image component is and thus what operation should be assigned to it. Note that many image segmentation methods, particularly those efficient ones tend to produce over-segmented results. Hence, the labelling is essentially a classification, which groups the over-segmented components produced by image segmentation. The classification is guided by semantic information attached to the objects. For example, a histogram is a common method to computationally represent some semantic information. For a typical product label (see Fig. 5) with date, price and/or other text on it, its histogram should be like the left image shown in Fig. 6. It can be observed that most of the pixels are white while a small group of pixels are black. For a typical barcode (see Fig. 5), its histogram should be similar to the right image shown in Fig. 6. If the system finds that the histogram of one component is similar to it, it will label this component as ‘barcode’.

Besides histograms, other statistical information can also be used to help the system analyse and understand image components in an intelligent manner. For example, the intensity standard deviation of the barcode and the label should be much higher than that of any other component.

Once the system understands the classified components, it can then assign different operations to each of them. For instance, for the component recognised as ‘date tick box’, the system can perform character recognition on it; for the component recognised as ‘seal’, the system can assign an algorithm to measure its minimum/maximum width; for the component recognised as ‘barcode’, the system can focus on the corresponding region and assign a barcode reader to it. Operations on different components could be carried out using different imaging modalities – for example, pack seal detection and analysis could be carried out using laser scatter imaging, food inside a pack could be analysed using 3D imaging, etc.

3.3. Training

It can be seen that two places in the workflow need training. In the image segmentation stage, as the outcome is sensitive to input parameters (it is also common that different types of images require different parameter settings), the system needs to be trained based on our own ground truth data to select proper parameters. Another place is semantic labelling where we need to classify image components using semantic information. The training phase at this stage is to tell the system what semantic information or features are desired (they also need to be extracted from our own ground truth data) and thus the classification can be encouraged to produce components with the desired semantics.

Additionally it could also be possible to combine both image segmentation and semantic labelling in a single step (so-called “semantic segmentation”), taking advantage of learned semantic information to improve segmentation and vice versa. This could lead to a more computationally efficient as well as more robust approach.

4. Future work and other issues

The planned work in the near future is listed as below:

- We shall investigate more efficient segmentation methods such as SLIC [2] and how to integrate this with a graph-based method to deliver a more efficient and reliable segmentation. SLIC is good at preserving

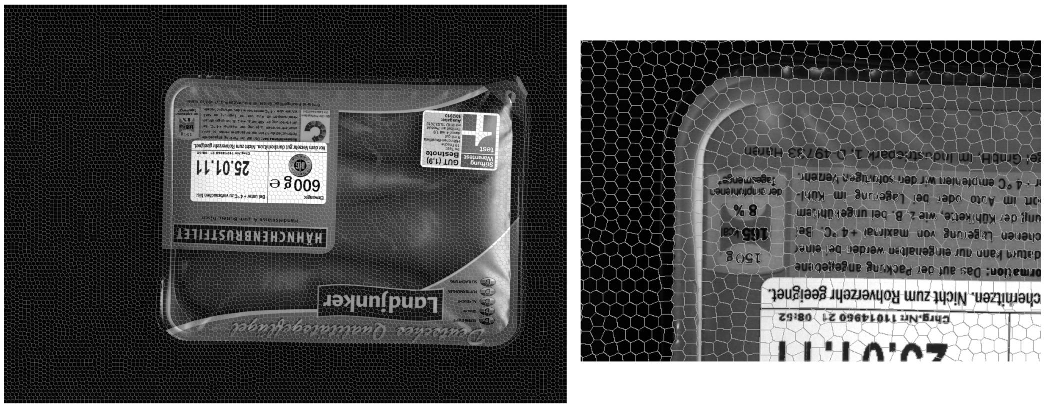


Figure 7: The implementation of SLIC on one of our test images using VLFeat C library and MATLAB

local details as shown in Fig. 7 and obviously the graph-based method using downscaled images can be used for the global background segmentation. We believe that the segmentation can benefit from such 'global-local' integration in terms of both efficiency and accuracy.

- We shall investigate different image labelling/classification methods such as Markov/Conditional Random Field-based methods and compared it with the Adaboost-based method [3] using our specific test images and ground truth data.

- Inevitably, we need to construct a database with sufficient images to run training and testing. And we also need to construct ground truth data for each image (like the MSRC labelled image databases[2]). This could be a lengthy task although we can first construct a small-scale database for preliminary research, and get help from the industrial partners in the project with obtaining real-world image datasets.

- Investigate using colour images. The work in July 2013 is completely based on greyscale images since it seems that currently, the camera mounted in the Ishida QC unit can only produce greyscale images. Although greyscale images show the potential to "get the job done" to some extent, we believe that colour images can make the system more

[2]http://research.microsoft.com/en-us/projects/objectclassrecognition

reliable simply because they are more informative. Specifically, for automatically detecting whether a food product is good or not (e.g., whether a tray of beef mince goes bad), it is very important, if not inevitable, to evaluate colour information of the product offered by colour images.

- We will also investigate using a 3D camera/scanner. The proposed system handles 2D images, while in a real world scenario we would have to consider the entire surface area and the shape structure of the 3D models of food products.

- Additionally, we will need to investigate (together with Ishida) the possibility of re-using some of the core components of the Sherlock system, including character recognition and barcode recognition, to avoid "re-inventing any wheels" and make best use of existing resources. We suggest that on this issue, the new vision software to be developed in the project should include the extraction of the group of black (or other coloured) pixels corresponding to the characters, and then the pre-existing software libraries for character recognition etc. would be applied.

References

[1] P. F. Felzenszwalb, D. P. Huttenlocher, International Journal of Computer Vision 59 (2004) 167–181.

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Susstrunk, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (2012) 2274–2281.

[3] M. Barnes, T. Duckett, G. Cielniak, G. Stroud, G. Harper, Journal of Food Engineering 98 (2010) 339–346.