LIVEITUP! 2 SMART REFRIGERATOR: IMPROVING INVENTORY IDENTIFICATION AND RECOGNITION

Juan Karlos P. Aranilla [1,2], Terence Anton C. Dela Fuente [1,2], Tonny York Quintos [1,2], Edmandie O. Samonte [1,2], Joel P. Ilao [2], Francis P. Lai [2]
[1] Center for Empathic Human-Computer Interactions, College of Computer Studies, De La Salle University, 2401 Taft Avenue, 1004 Manila, Philippines
[2] Computer Technology Department, College of Computer Studies, De La Salle University, 2401 Taft Avenue, 1004 Manila, Philippines

Abstract: In recent years, the consuming public has seen the introduction and steady rise in popularity of appliances that are greener and more user-interactive. These so-called "smart appliances" give a long term economic advantage, but they are more expensive than standard appliances and consumers are still hesitant to replace their old working units. An alternative and economical approach would be to fit regular working units with sensors, in order to approximate functionalities of their "smart" counterparts. To this end, we have developed a prototype called the LiveItUp! Smart Refrigerator - a standard refrigerator that is fitted with a host of sensors and an image processing software for inventory and item recognition [3]. The system added to the resulted to less power consumption and made the fridge capable of inventory tracking. However, one of the limitations of the smart fridge is the software used for image processing; it can only detect a single item and object recognition was done via template matching which was computationally expensive and at times caused the image processing software to crash.

In this paper, we describe LiveItUp! 2 Smart Refrigerator, which is an improvement in vision-based object detection and recognition of food inventory for the previous smart fridge. Different image processing techniques such as motion detection, Thresholded Euclidean Difference, blob processing and cropping are used. The system is now capable of identifying multiple items and the image processing application can handle object identification and recognition without crashing.

Key Words: motion detection; Thresholded Euclidean Difference; blob processing; cropping;

1. INTRODUCTION

The technology industry and various research facilities have geared towards on developing smart homes, and smart appliances [2,5,6,9,10]. The industry attempts to change the traditional fridge function, i.e., to store food items in a cool environment, to integrating fridge with TV, radio, and computer capabilities, LCD screens and even connection to the Internet. A smart fridge must be able to track items and to lessen power consumption. It is a well-known fact that the fast-paced development and modern living has resulted in a change of people lifestyle towards less time to keep track of food inside the fridge [1] and since the fridge is one of the appliances that consumes a lot of energy, there is a need to make it efficient.

To realize the smart fridge, the LiveItUp! Ecofriendly Smart Refrigerator has been developed. It is a standard refrigerator that is fitted with different sensors such as optical sensors, humidity sensors, RFID scanner and temperature sensors. Through the use of the RFID scanner, the system can identify the user who comes in contact with the fridge. The optical sensor is used for capturing images inside the fridge and an image processing application was developed to process its data and identify the items inside the fridge. The humidity and temperature sensors were used as a feedback in controlling the compressor of the fridge. The smart fridge has been able to detect items and also lessen the power consumption.

Although the LiveItUp! Fridge has realized that standard fridges can become smart in terms of tracking the inventory and lessening the power consumption, there were still some improvements to be done to the image processing application. One of the limitations of the system was that it cannot detect multiple items. Also, object recognition was done by template matching that made the image processing application to crash. Therefore the LiveItUp! 2 is developed to improve the object identification and recognition of the previous system.

2. SCOPE AND LIMITATIONS

This research concentrates on improving the object identification and recognition capabilities of the system. The image processing software that we will improve must be able to detect multiple objects inside the fridge and the user must be able to tag it. When the object is detected again, the software must be able to recognize the object.

The data that we will be working on is a video stream of what the user is doing inside the fridge and then different images will be captured in the video stream. Figure 1 (left) shows the area of the fridge where we will record the data. This area will be recorded by a camera that is placed outside of the refrigerator. This research will use the first prototype and use a single camera. The user can add, move and remove an object one at a time.

Figure 1. Area of interest marked in red (left) and Physical setup of the system (right).

3. SYSTEM OVERVIEW

The system is composed of three main components namely the refrigerator, processing unit, and the optical sensor. Figure 1 (right) shows how the system is setup. The refrigerator holds the items that the system will detect. The optical sensor, on the other hand, is responsible for capturing the video stream of the fridge compartment. Lastly, the processing unit is responsible for acquiring and processing the video stream.

A. Refrigerator

The refrigerator currently used is the first prototype. All of the sensors for user identification, door, humidity and temperature are not installed. Only the middle compartment is used as the area in which the optical sensor is going to capture video streams.

B. Processing Unit

The processing unit is responsible for applying image processing to the acquired video stream. The processing unit applies image processing to the video stream in six stages as shown in Figure 2.

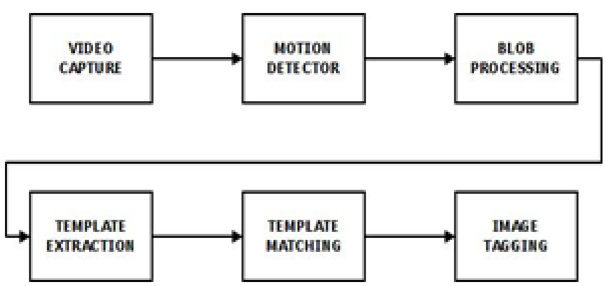

Figure 2. Block diagram of the system.

At first the processing unit will acquire the video stream from the optical sensor. The first thing that the image processing software will do is to capture a background image from the video stream. At the start-up of the software it will detect movement from the video stream since it will detect a change in brightness (when the optical sensor is still off there is no light present and when the optical sensor is turned on there will be light present. This transition from off to on and change in brightness will trigger a movement and will be detected by the Motion Detector). At this point the system assumes that there are no items inside the fridge compartment therefore it will only capture the background of the fridge compartment. Figure 3 (left) describes the process. Figure 4 (top) shows an example of a background image.

If there is no user activity such as adding, moving or removing an item inside the fridge compartment, the application will stay idle. The Motion Detector will check if there is still movement. When the user is done and the Motion Detector has not detected any activity anymore, it will capture an image and save two copies of the image. The application will now do Thresholded Euclidean Difference to the background image and to one of the copies of the image. The other copy will be stored as a source image. The resulting image is a binary image and only blobs could be seen. The resulting image will go through blob processing by filling the holes of the blobs, removing the small blobs and counting the number of blobs present in the image. The blobs will be extracted through image cropping. The cropped image will be changed back to its original colored image and will be saved as a template. The filename of the template is saved by its x and y coordinate in the original image and height and width of the template.

If there would be another user activity, the same process will happen but the new input for the Thresholded Euclidean Distance is the past image (it may not be the background image anymore but the current image before another user activity) and the newly acquired image (image after the Motion Detector has detected an activity). This is the method used for extracting item signatures in an image. Figure 4 (bottom left and bottom right) shows an example of an extracted template and a source image.

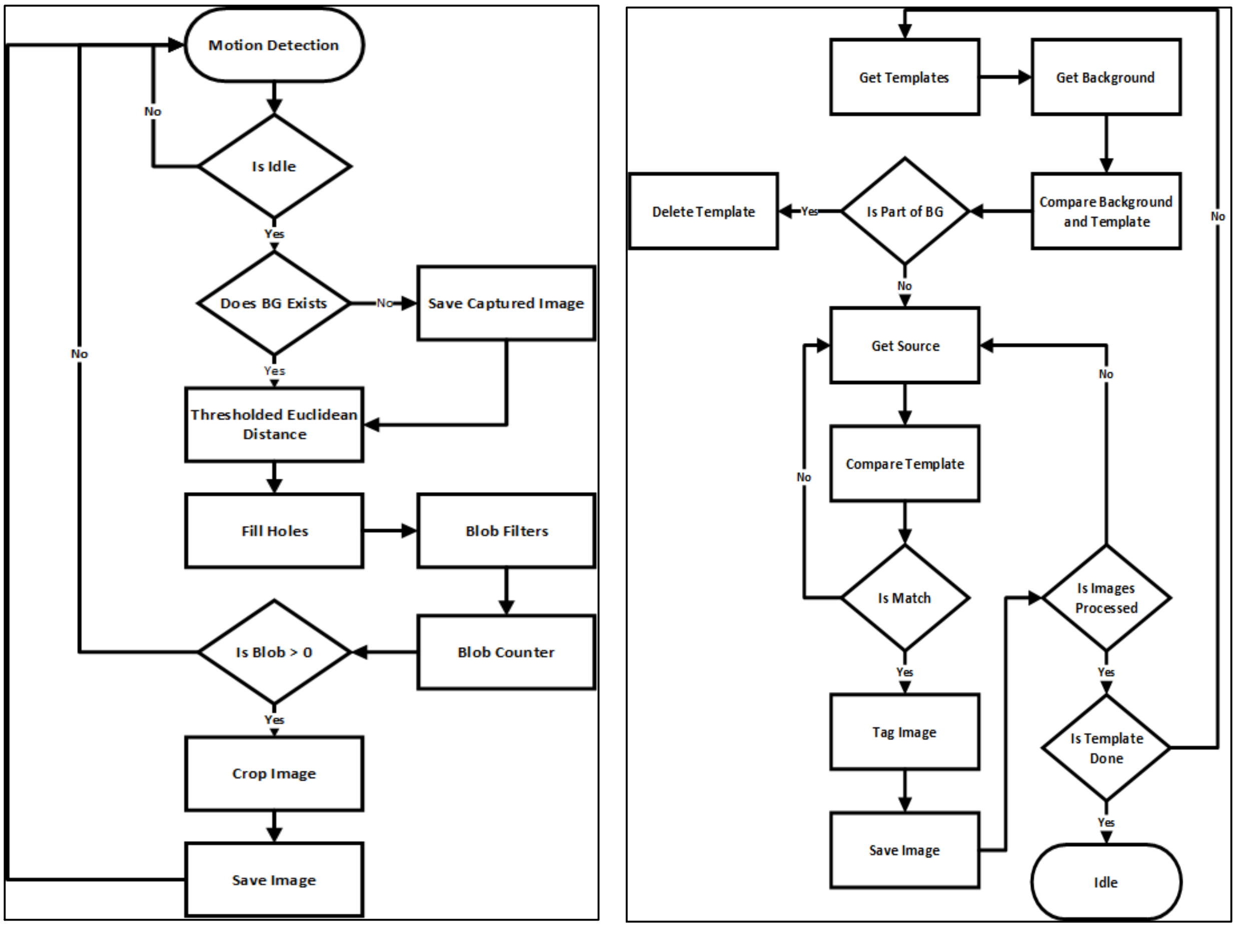

Figure 3. Motion detection, pre-processing and template extraction (left)
and Item identification and recognition (right).

When all the templates are extracted, the image processing application will now identify and recognize the items. First it will get a template and the background image. It will compare whether the template is part of the background image. If it is, the template will be deleted. This just means that the template that was extracted has no item in it. If the template is not part of the background, then it will be compared to the source image. Using exhaustive template matching, the template will be matched to the source image. If the template can be found in the source image, the item in the source image will be tagged. After tagging the item, the application will check for another template and it will do the process again. Figure 3 (right) describes the process.

Figure 4. An example of a background image (top),
an example of a template (bottom left) and a source image(bottom right).

C. Optical Sensor

The optical sensor that we are using is an A4Tech Web Camera. It can capture images and video streams.

4. RESULTS AND DISCUSSION

We tested our image processing application by detecting a pizza. First the application will detect the background. The background is the fridge compartment where we place the pizza. This is shown in Figure 5 (left).

Figure 5. Background image (left) and Pizza template (right)

For the first scenario, we put the pizza in the fridge compartment. The application will detect the pizza, store the template of the pizza and store the whole image. This is shown in Figure 5 (right) and Figure 6 (left).

The application will process the template and the source image. The output image is showed in Figure 6 (right). Unfortunately due to the limited coverage of the optical sensor, the filename of the template is not shown in the output image. Yet it is still cropped.

We also tested if the image processing software can detect multiple objects. We added other items such as a Cheez Whiz Spread and a circular Tupperware. Figure 7 shows their respective templates. The image processing application detected all of the items in the source image shown in Figure 8 (top left) but it made output images, each having only one item tag. This is shown in Figures 8 (top right, bottom left and bottom right).

Figure 6. Source (Whole) Image (left) and Item Identification and Recognition (right).

Figure 7. Cheez Whiz (left) and Circular Tupperware (right) template.

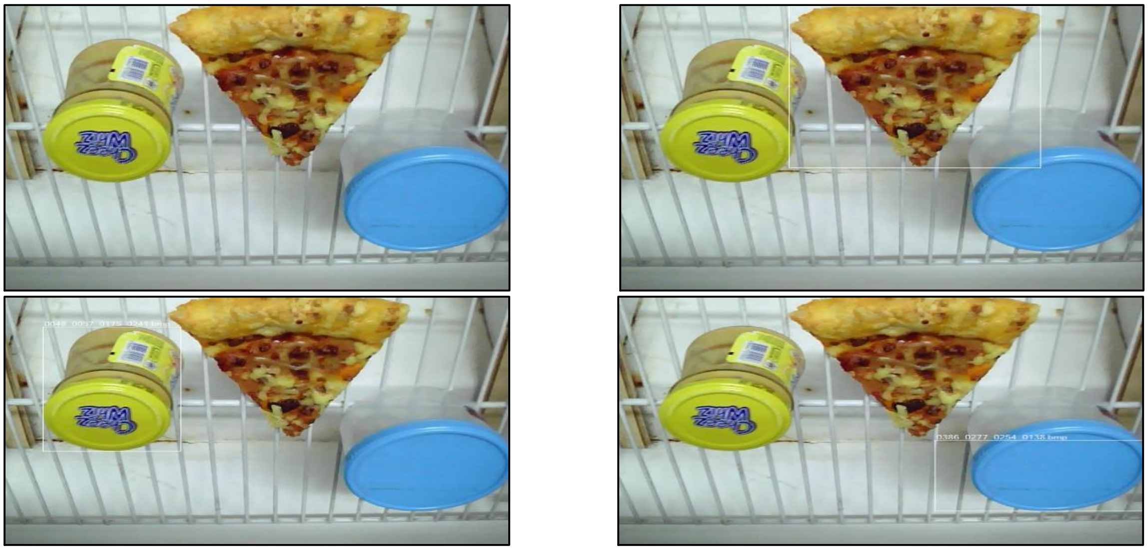

Figure 8. Source image having three items (top left), pizza (top right), Cheez Whiz spread (bottom left) and circular Tupperware (bottom right) are all identified and recognized.

5. ON-GOING WORK AND RECOMMENDATION

The template matching would be improved, the processing speed must be faster or a new template matching algorithm must be used. An alternative would be Fast Template Matching or Adaptive Template Matching Threshold. Another improvement would be the use of shadow removal techniques. This is due to the fact that when the application gets a template, it also gets the shadow. Another improvement would be image repositioning. Sometimes an item cannot be tagged if the template is oriented differently even if the template is the same with the item. The application also makes more than one template of the item when there is a shadow or the orientation is not the same. Instead of using shadow removal or image repositioning, we can use the naming convention we give for a template which is the x and y coordinate and width and height of the template. If two templates have almost the same name, we can delete one of the template. If we still pursue developing a shadow removal and image repositioning algorithm, the application can get better templates and can better tag the items. Template extraction and item tagging techniques can also be improved by acquiring templates and clustering them by classes and not extracting and tagging specific items instead. To further improve the system, machine learning is being considered to be integrated into the system as a means to improve item recognition and to test the similarities of each of the extracted templates.

6. CONCLUSION

The image processing application can detect any object. It can be used to detect single or multiple items. But for multiple item detection, the processing slower compared to single object detection and the image processing application generates multiple output images with just one item tag for each of the output images. The application can also recognize the item by tagging it using the template's file name. However, the processing time for template extraction is slow and if there are multiple objects, the processing time becomes slower.

7. REFERENCES

[1] C. E. Woteki and P. R. Thomas, Eat For Life: The Food and Nutrition Board's Guide to Reducing Your Risk of Chronic Disease Washington, D.C. National Academies Press, 1992.
[2] Chan, M., Esteve, D., Escriba, C., Campo, E., Review of smart homes—Present state and future challenges, Computer Methods and Programs in Biomedicine, Volume 91, Issue 1, July 2008, pp. 55-81.
[3] Chua, J., Dela Fuente, T. Lai, F., Magpantay, A., Regalado, R., Samonte, E. (2011). "LiveItUp! Smart Refrigerator", Science and Technology Conference, De La Salle University.
[4] D. Christianson, Home smart home, http://www.researchworks.carleton.ca/2005_Spring/79.htm, 2005
[5] ElecVision, Smart Home Solutions, www.elecvision.com.au, 2006
[6] L. Cerwinske, Smart Home Design, Reed Press, 2004
[7] L. Jiang, and B. Yang, , Smart Home Research, Proceedings of the IEEE International Conference on Machine Learning and Cybernetics, Shanghai, 26-29 August 2004.
[8] N. King, Smart Home-What do consumers want?, Intertek Research&Test Centre, 2003.
[9] R. Beuran, J. Nakata, T. Okada, T. Miyachi, K. Chinen, Y. Tan, Y. Shinoda (2007), Performance Assessment of Ubiquitous Networked Systems, 5th International Conference on Smart Homes and Health Telematics (ICOST2007), Nara, Japan, June 21-23, 2007, pp. 19-26.
[10] R. C..Elsenpeter, T. J.Velte (2003), Find in a Library: Build your own smart home, McGraw-Hill