An Image Processing and Pattern Analysis Approach for Food Recognition

By

Parisa Pouladzadeh

A thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of

Masters of Applied Science

School of Electrical Engineering and Computer Science

University of Ottawa

Abstract

As people across the globe are becoming more interested in watching their weight, eating more healthily, and avoiding obesity, a system that can measure calories and nutrition in everyday meals can be very useful. Recently, there has been an increase in the usage of personal mobile technology such as smartphones or tablets, which users carry with them practically all the time. In this paper, we proposed a food calorie and nutrition measurement system that can help patients and dieticians to measure and manage daily food intake. Our system is built on food image processing and uses nutritional fact tables. Via a special calibration technique, our system uses the built-in camera of such mobile devices and records a photo of the food before and after eating it in order to measure the consumption of calorie and nutrient components. The proposed algorithm used color, texture and contour segmentation and extracted important features such as shape, color, size and texture. Using various combinations of these features and applying a support vector machine as a classifier, a good classification was achieved and simulation results show that the algorithm recognizes food categories with an accuracy rate of 92.2%, on average.

Acknowledgments

First and foremost I would like to express my deepest gratitude to my supervisor, Professor Shervin Shirmohammadi, for assigning me to this project which is intended to help people with diabetes and to improve health systems. Without his support, the completion of this thesis would not be possible.

I had the opportunity to work as part of a great team during the course of my thesis. The people involved in the project gave me a better perspective and new knowledge. I would like to thank all of my colleagues in the DISCOVER lab, especially Gregorio Villalobos, for providing valuable information and help.

Finally, I owe my loving thankfulness to my family and I thank my husband, Mehdi, for his kind help and patience.

Table of Contents

List of Figures

List of Tables

Glossary of Terms

2D	Two Dimensional Space
3D	Three Dimensional Space
BLOB	Binary Large Object
DLW	Doubly Labeled Water
FFQ	Food Frequency Questionnaire
HLS	Hue, Lightness and Saturation
HSI	Hue, Saturation and Intensity
HSV	Hue, Saturation and Value
L-a-b Model	Lightness and distribution dimensional color distribution over a and b
NIMS	Nutrient Intake Monitoring System
PDE	Partial Differential Equation
RFID	Radio Frequency Identification
RBF	Radial Basis Function
RGB	Red, Green and Blue Color Space
RGBA	Red, Green, Blue and Alpha Color Space
ROI	Region of Interest
SVM	Support Vector Machine
WHO	World Health Organization
YCrCb	Digital Video signal for Luminance and dimensional color distribution over Cr and Cb
YIQ	Analog Video signal for Luminance and dimensional color distribution over I and Q
YUV	Analog Video signal for Luminance and dimensional color distribution over U and V

Chapter 1 Introduction

1.1 Motivation

The increasing number of obesity patients among young people is a huge problem these days. Unfortunately, this trend is continuing to grow [1] and has also been linked to an increase in type 2 diabetes [2]. In 2008, one in ten of the world's adult population was obese [1], while in 2012 this figure had risen to one in six adults [2], an alarming growth rate. Recent studies have shown that obese people are more likely to have serious health conditions such as hypertension, heart attack, type 2 diabetes, high cholesterol, breast and colon cancer, and breathing disorders. The main cause of obesity is the imbalance between the amount of food intake and energy consumed by individuals [3]. So, in order to lose weight in a healthy way, as well as to maintain a healthy weight for normal people, daily food intake must be measured [4]. In fact, all existing obesity treatment techniques require the patient to record all food intake each day in order to compare food intake to energy consumed. Unfortunately, in most cases, patients face difficulties in estimating and measuring the amount of food intake due to self-denial aspects of the problem, lack of nutritional information, the manual process of writing down this information (which is tiresome and can be forgotten), and other reasons.

Dietary intake, the process of defining what someone eats during the day, provides valuable insights for increasing intervention programs which leads to preventing many diseases. As will be explained in chapter 2, accurate dietary intake measurement is considered to be an open research problem in nutrition and health fields.

In this research we introduce a new semi-automatic dietary assessment application which assists dieticians in monitoring daily nutrient intake. To do so, we have engaged various image processing techniques in different steps of our food recognition system. By hiring this food recognition method on their mobile devices, any individual will be able to monitor his/her food intake.

A high level block diagram of the steps for this research is shown in Figure 1-1.

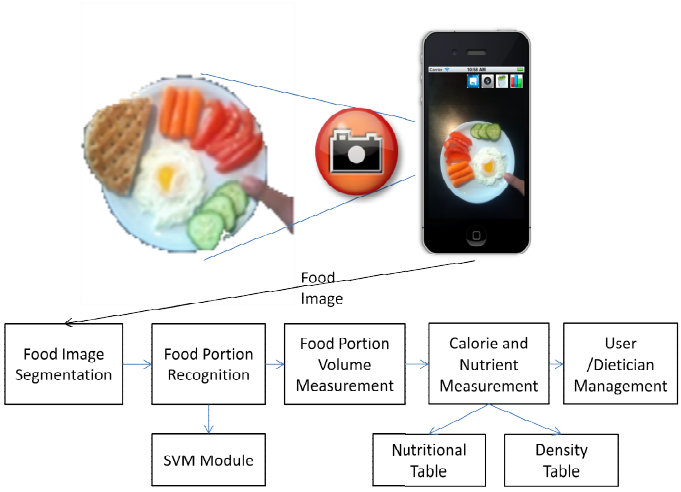


Figure 1-1 Overall system design.

As the figure shows, in the first step a picture will be taken of the food dish. Using image segmentation techniques (e.g. color segmentation and texture segmentation), the food portion areas will be extracted. Next, food portions will be classified using food recognition techniques (e.g. a Support Vector Machine). In the next step, the volume of each detected food portion will be measured. Then, the calorie information for each recognized food portion will be estimated according to its volume and based on nutritional tables. Finally, the extracted calorie information will be sent to the user.

As shown in Figure 1-1, our dietary assessment system is composed of several independent steps, where the outcome of one step is the input for another. Hence, the overall intake measurement is

very sensitive to the operation of each process. In other words, a mis-estimation in an early step, such as the segmentation phase, will propagate through the whole system and reduce the final accuracy. In this project, we have taken this characteristic into consideration in order to have a precise intake measurement system. However, we have to qualify that by “precise”, we mean the technical accuracy of our individual algorithms. The precise or accurate measurement of the overall food calorie is another issue that we will discuss next.

1.2 Accuracy Objectives

Before discussing any technical issues, it is important to understand what level of accuracy is expected from our system. To answer this question, we must first see what level of accuracy existing clinical methods have in their measurement of food’s nutritional facts. There are two things to consider. First, if we put a plate of food in front of an expert dietician, s/he cannot give an accurate measurement of its nutritional facts by simply looking at it or even examining it manually, because it is impossible to know the exact contents of the dish, such as whether this dish contains salt, and if so how much, or whether it contains oil, and if so what type (olive, corn, animal-based, …), and if so how much, etc. Also, some food portions can be obscured, for example a piece of meat could be deep inside a soup, making it invisible to the dietician. So we can see already that highly accurate calorie measurement is not possible in real life. Second, when we add this to what happens in existing clinical methods such as [4] and [5], in which the dietician goes over a list of food items recorded by the patient (without necessarily even seeing the actual food or its picture, and without knowing the size of portions), it becomes clear that accuracy is decreased even more. We will see more about these existing methods and their shortcomings in chapter 2. This is very important, because it directly affects the goal of our system, described in the next section.

1.3 Thesis Goal

Our goal in this thesis is:

> *To design a semi-automated measurement tool running on smartphones or other mobile devices with a built-in camera that facilitates the recording of food calorie intake, compared to existing clinical methods, by measuring the size of food portions and their nutritional facts*.

Our goal is not to necessarily have high accuracy, because as explained above such accuracy is not possible in practice. Of course, the more accurate the system, the better the end results, and this is why in this thesis we have tried to measure the size of food portions and identify the food type as accurately as possible. But it is very important to understand that high accuracy is not possible when dealing with food pictures only.

1.4 Contributions of this Thesis

- Improving the segmentation phase of existing food recognition algorithms by enhancing image processing components with texture segmentation in addition to the color, size and shape features used in previous methods.
- Design and development of a support vector machine (SVM) specifically as a food classifier for labeling each food segment and appropriate configuration of the SVM's parameters for training and testing levels.
- Design of a food specific image database to recognize food area and feasibly recognize the overall food. The database has these features:
    - It contains simple training images with only one type of food in it, and we can use those images as patterns to perform recognition.

- It has images from each food at different illuminations.

- Implementation of a proof-of-concept and validation with performance evaluation.

1.5 Publications Resulting from this Work

Conference papers:

[1] G. Villalobos, R. Almaghrabi, **P. Pouladzadeh**, S. Shirmohammadi, "An Image Processing Approach for Calorie Intake Measurement," *IEEE International Conference on Medical Measurement and Applications (MeMeA)*, pp. 71–75, 2012.

[2] R. Almaghrabi, G. Villalobos, **P. Pouladzadeh**, S. Shirmohammadi, "A Novel Method for Measuring Nutrition Intake Based on Food Image," *IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pp. 366–370, 2012.

[3] **P. Pouladzadeh**, G. Villalobos, R. Almaghrabi, S. Shirmohammadi, "A Novel SVM Based Food Recognition Method for Calorie Measurement Applications" *IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, pp. 495–498, 2012.

Journal paper (submitted):

[1] S. Shirmohammadi, **P. Pouladzadeh**, R. Almaghrabi, "Measuring Calorie and Nutrition Amounts from Food Images," *submitted to IEEE Transactions on Instrumentation and Measurement Journal*, 2012.

1.6 Thesis Organization

The organization of the thesis is as follows. In chapter 2, some of the related works are explained. The technology background and various design tools are explained in chapter 3. In chapter 4, the proposed system for food recognition is depicted. Chapters 5 and 6 represent the implementation and evaluation steps of the thesis. Finally the thesis concludes in chapter 7.

Chapter 2 Related Work

Considering the popularity of mobile devices (e.g. smartphones and tablets) among young people, and the ubiquity of these communication devices, they can be used as a viable target for including dietary related applications, such as our food recognition system.

This research aims to show the helpfulness of this technology in solving one of our real life problems, which is food intake measurement. The current research in this field can be classified into three categories, namely clinical, assistant-based, and semi-automated approaches. In clinical research, such as the 24-hour dietary recall (24HR), and food frequency questionnaire (FFQ), data are written manually by the patient and calorie intake will be extracted afterwards. With these methods, a high number of errors can happen in the overall process since the data are written manually. Another drawback with these methods is how difficult it is to make the patient commit to the treatment. Recently, some researchers have worked on semi-automated methods. Our project will develop a semi-automatic application, with a simple but functional design, to encourage people to interact with the application. Furthermore, the target hardware is mobile devices, such as smart phones, so in this case we intend to make use of new technology to increase the chances of use.

In the following section, there is more explanation about the three methods for calorie measurement.

2.1 Clinical Approaches

In this section we review current clinical approaches for calorie-intake measurement, as well as their advantages and disadvantages.

2.1.1 24-Hour Dietary Recall

With 24-hour dietary recall, the respondent is asked to remember and report all food consumed in the previous 24 hours. The recall is typically conducted by interview, in person or by telephone [3]. If possible, interviewers should be dieticians with education in food and nutrition; however, non-nutritionists who have been trained in the use of a standardized instrument can be effective. The interview usually needs specific probes to help the respondent remember all foods consumed throughout the day. An early study found that respondents who checked their reports with an interviewer and received their comments regularly got 25% better results than respondents without an interviewer investigating. This means that self-prevention of eating only is not enough for preventing obesity, investigating actual calorie consumption is necessary. In this method, having the interviewer investigating daily reports helped to get a better program for the other days [4]. The Food Surveys Research Group (FSRG) of the United States Department of Agriculture (USDA) has devoted considerable effort to improve the accuracy of this method, but the major drawback of the 24HR method is the issue of underreporting food consumption [5].

Features such as obesity, gender, restraint, education, literacy, perceived health status, age and ethnicity have been shown to be related to underreporting [6]. Harnack et al. [7] found important underreporting of large food portions when food models showing recommended serving sizes were used as visual aids for respondents. There are also other studies conducted that focus on underreporting of food intake, in order to define a pattern in this problem present in obesity treatment [5], [8]. The advantages and disadvantages of this method can be categorized as follows.

Strengths:	Weaknesses:
• Low respondent burden • Suitable for large scale surveys • Can be administered by telephone	• Estimation of portion sizes • Single observation provides poor measure of individual intake • Bias in recording "good/bad" food, memory dependent

2.1.2 Food Frequency

The food frequency approach [9] asks respondents to report their usual frequency of consumption of each food from a list of foods for a specific period of time. Information is collected on frequency and sometimes portion size, but little detail is collected on other characteristics of the foods, such as the methods of cooking or the combinations of foods in meals. To estimate relative or absolute nutrient intakes, many FFQs also integrate portion size questions, or specify portion sizes as part of each question.

The estimated food diary is a prospective dietary assessment method which provides detailed data on food and nutrient intakes. Individuals record detailed information of consumed foods and beverages and this information includes brand name, cooking and preparation method, portion sizes and so on. Foods can also be weighed and recorded if an individual chooses, but the main purpose of using this method is to avoid the burden of weighing. Traditionally the method is pen and paper based but the food diary may also be completed by digital recording. At the end of the assessment period, ideally a trained interviewer should go through the record with the individual to clarify details – this could be done by telephone. A short questionnaire can be included to aid interpretation of the record and to provide details of core foods regularly eaten (e.g. type of milk) and to inquire about non-food items such as dietary supplements. Parents can complete diaries for

young children or in addition to children in older age groups. Individuals may record the time, location and whether the respondent was alone or with others for each eating occasion, thus providing information on eating patterns and the social context of eating. We have categorized the pros and cons of this method as follows.

Strengths:	**Weaknesses:**
• Low respondent burden • Suitable for large scale surveys • Can be self-completed • Can be posted	• Estimation of portion sizes (though use of food photographs may improve precision) • Possible over-reporting of 'healthy' foods (e.g. fruit and vegetables) • Needs to be validated in relation to reference method

For both of the above explained methods, data are listed manually, so this may cause some errors in the overall process. Another disadvantage is the difficulty in getting patients to commit to the treatment and encouraging them to take care of their nutritional intake.

Our project will develop a semi-automatic application, with a simple but functional design, to encourage the patient to interact with the application. Furthermore, this project should be more practical through developing the software application for mobile devices (such as smartphones, tablets and netbooks).

2.2 Assistant-based Approaches

Nowadays, by applying technology to daily affairs, a mobile-based calorie measurement is created. With this method, people can use mobile devices as a user-interface and send their data online to the specialist to calculate the amount of the food in the image. Mobile applications provide a

unique mechanism for gathering dietary information which reduces the burden on record keepers. A method which people use on their mobile telephones nowadays is shown in Figure 2-1 [6].

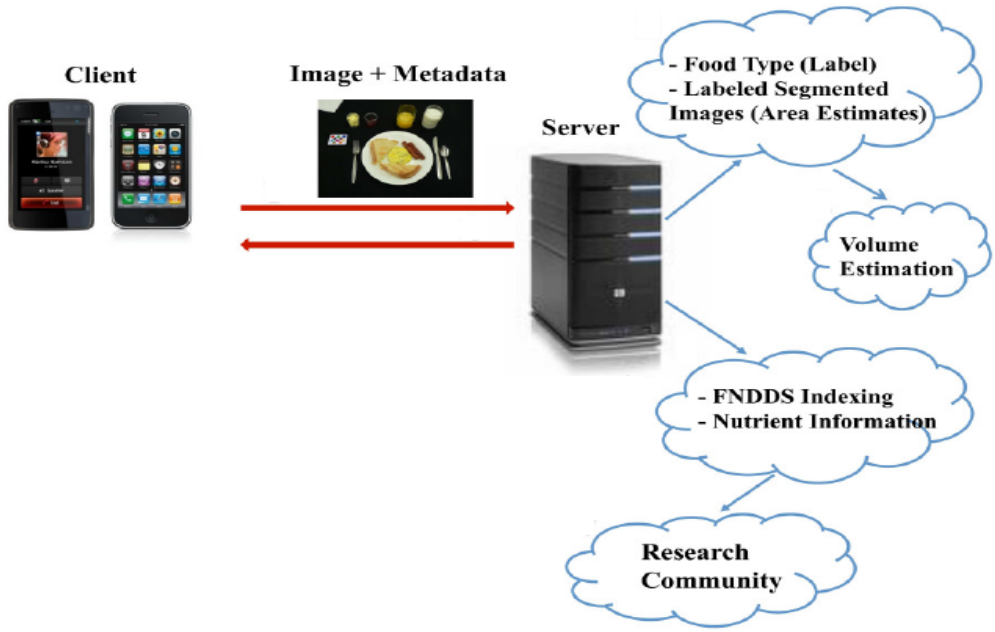


Figure 2-1 Architecture of the mobile device application [6].

As the figure shows, the first step is to send the captured image to the server for manual analysis. The nutrient information is extracted from the food using the USDA Food and Nutrient Database for Dietary Studies (FNDDS) database [10]. FNDDS is a database of nutrient values contained in food eaten in the U.S. With this method, after taking an image from the food, the image will be sent for particular processing to the server and every calculation sent over the server will be done manually. After that, analysis results are sent back to the user where the user confirms and/or adjusts this information.

2.3 Semi-automated Approaches

Another method which can work more accurately and faster than the previous ones, and which could replace them, is the semi-automated approach. Firstly, the interaction of food analysis based on imaging processing with the help of high end technologies has been proposed by Harnack et al. [7]; in this research the image analysis is focused on large food portions. Sun et al. [11] proposed a similar approach, where the idea is to take pictures of food, then, based on a calibration card located inside the picture as a measurement pattern, the size of food portions are calculated, the food is manually identified with the help of nutritional information retrieved from a database, calories are calculated for each picture and finally the complete set of information is stored in a different database in the research facility. In this case, based on the known size of the calibration card, portions can be translated into real-life sizes, and the calculations are closely related to the real caloric content of each food. Martin et al. [12] proposed a system where the user captures the images, also with a calibration card, then the image must be sent to a research center to be analyzed. This, of course, comes with its own shortcomings in terms of offline data processing. Meanwhile, our system is intended to perform analysis in the place where the patient is located.

Using a different approach, Chi et al. [13] introduced the idea of a smart kitchen, designed entirely to aid a normal person to be aware of the calories associated with the food being prepared inside this facility. An overhead camera captures the images during the preparation process; meanwhile weighing sensors are working in other sections of the measuring control – in the counter and the stove respectively. The environment is an augmented reality, where normal tasks performed in the kitchen are enhanced with different sensors. Information is processed by the application, and the user can obtain immediate responses from the application. In a similar approach, Westerterp-

Plantenga [14] designed a food intake monitoring system, where the complete set of analysis involves a special table where the plate is located while the patient is eating. The table constantly measures the weight of the food, and the information is quantified in order to measure how much is consumed from the original portion, what is the speed of the patient in the process, and what is the gap between bites. In addition, the weight of each bite is considered in order to measure the differences between bites. All this information is compiled in the research laboratory, and the patient must be controlled for a period of several hours. The user typically does not like this kind of control and its attached environments, is prone to forget or prefers not to use this type of measurement procedure, as weighing systems are not available everywhere and are not easy to carry around. Chang et al. [15] proposed a dining table equipped with different sensors located within the table and Radio Frequency Identifier (RFID) tags. The RFID identifies the food and the sensors weigh the amount of the portion. But since this requires the user to eat only at that table, such environments are impractical if we intend to apply this kind of measurement in common situations such as food courts or restaurants.

Another method that has used an image-based analysis system is described in [16], and in this method, users capture food images using a mobile phone camera. Based on information (i.e., food name and code) determined through food segmentation and classification of food images, this system chooses a particular food template shape corresponding to each segmented food. Finally, the system reconstructs the three-dimensional properties of the food shape from a single image by extracting feature points in order to size the food shape template. By employing this template-based approach, the system automatically estimates food portion size, providing a consistent method for the estimation of food volume. Also in [17], a method which automatically determines regions in an image and identifies the food type based on its features is explained. This method

used a Normalized Cuts segmentation based on intensity and color. Furthermore, another method based on image segmentation results is proposed in [18]. In this paper, the computer vision strategies used to recognize fruit rely on four basic features: intensity, color, shape and texture. In addition, [18] proposes only color and texture features for fruit recognition. The recognition is done by the classifier based on the statistical features derived from Wavelet transformed sub-bands.

In our proposed system, we also aim at using smartphones as monitoring tools as they are widely accessible and easy to use. We will take advantage of different image processing for our analysis. However, we will try to improve the quality of the processing compared to the existing literature. Moreover, using an accurate classification will lead to more acceptable results.

Chapter 3 Technology Background and Design Choices

In this chapter, we will briefly cover the technologies behind our system: image pre-processing, image segmentation, feature extraction, and machine-learning based classification. Then, in chapter 4, we will show how we used these technologies in our proposed system.

3.1 Image Pre-processing

Image pre-processing refers to operations that are performed to make images ready for further processing. This pre-processing operation includes color space conversion, image cropping, image padding, and so on. In this subsection we will review some of the pre-processing functions related to this project.

3.1.1 Color Spaces

Inside of the digital images, we have picture elements or pixels, these pixels contain finite values (three or four values) generally based on a mathematical abstraction model. These mathematical models are used to define the internal structure of the image, also called color spaces. Here, we have basic models such as RGB and RGBA. The RGB color model is an additive color model in which red, green and blue light are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green and blue.

The main purpose of the RGB color model is for the sensing, representation and display of images in electronic systems, such as televisions and computers. The RGB color model already has a solid theory behind it, based on human perception of colors. RGB is a device-dependent color model; different devices detect or reproduce a given RGB value differently since the color elements and their response to the individual R, G, and B levels contrast from maker to maker, or even in the same device over time. Thus an RGB value does not define the same color across devices without some kind of color management.

The International Commission of Illumination (CIE) has defined a specific diagram to define the color chromaticity defined in the plot, which is the wavelength of light in nanometers. Because this representation is a well-accepted standard, many instruments of color measurement, color display, and color capture use this definition to work with the wavelengths of each color in order to perform the corresponding analysis and representation in a two dimensional plot.

Figure 3-1 shows the CIE chromaticity diagram. The scale of each axis is in nanometers, based on the wavelengths of the colors moving in the RGB scheme.

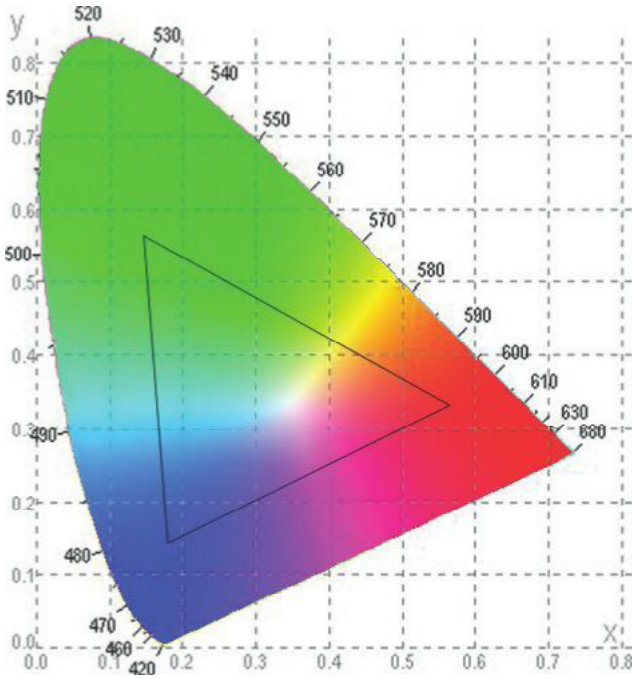


Figure 3-1 CIE chromaticity diagram.

There are two other color systems known as HSI (Hue, Saturation and Intensity), HSV (Hue, Saturation and Value) or HLS (Hue, Lightness and Saturation). These color spaces are highly related to how human vision works, and also how to represent the physical characteristics of color considerations from many different specimens. There is a specific relationship between the color spaces in HSI and RGB, and with the combination of both, a simple conversion of RGB channels can be translated into a specific new color space, highly dependent on the primary information generated by the RBG color space, known as YCrCb. Intensity falls into the characterization of Y, while hue and saturation are kept in the chrominance space corresponding to Cr and Cb.

Figure 3-2 shows a bi-conic representation of the Hue, Saturation and Intensity color space. As shown in this figure, the distance between black and white defines the intensity value, saturation is defined by the amount of color used and hue is the color itself. With the value of hue, we can have a unique numerical representation for each color on the space representation.

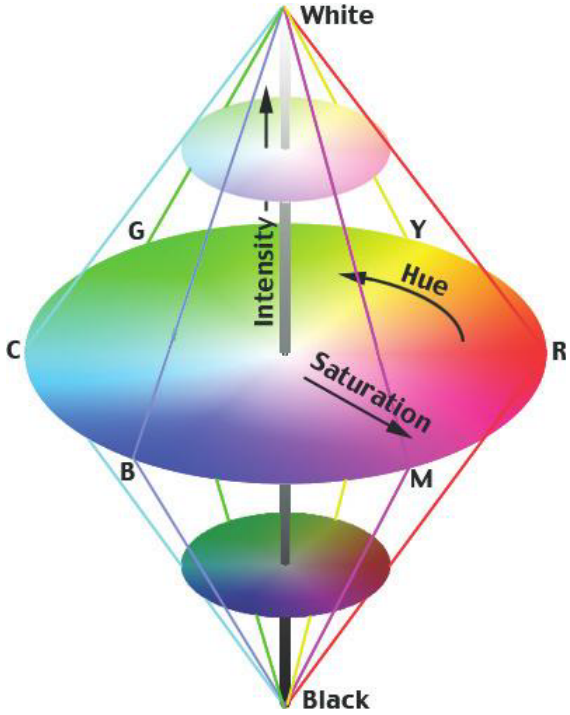


Figure 3-2 Hue-Saturation-Intensity space bi-conic representation.

3.1.2 Conversion between RGB and HSI

The RGB and HSI color spaces can be simply converted to each other by using some mathematical formulas and performing some matrix multiplication. In the following subsections we will explain the steps for converting these two color spaces.

3.1.2.1 Converting from RGB to HSI

To move from RGB to HSI, first using formula (1), I, V_1 and V_2 components will be generated for each of the image pixels.

$$\begin{bmatrix} I \\ V_1 \\ V_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{-1}{\sqrt{6}} & \frac{-1}{\sqrt{6}} & \frac{2}{\sqrt{6}} \\ \frac{1}{\sqrt{6}} & \frac{-1}{\sqrt{6}} & 0 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (1)$$

In order to generate H and S component formulas (2) and (3) have to be performed on V_1 and V_2.

$$H = \arctan\left(\frac{V_1}{V_2}\right) \tag{2}$$

$$S = (V_1^2 V_2^2)^{1/2} \tag{3}$$

3.1.2.2 Converting from HSI to RGB

In order to convert from HSI to RGB color space, we first extract the V_1 and V_2 values for each image unit as shown in formulas (4) and (5).

$$V_1 = S\cos(H) \tag{4}$$

$$V_2 = S\sin(H) \tag{5}$$

Having the values of V_1 and V_2, the R, G and B component can be generated using the transformation introduced in formula (6).

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & \frac{-\sqrt{6}}{6} & \frac{\sqrt{6}}{2} \\ 1 & \frac{\sqrt{6}}{6} & \frac{\sqrt{6}}{2} \\ 1 & \frac{\sqrt{6}}{3} & 0 \end{bmatrix} \begin{bmatrix} I \\ V_1 \\ V_2 \end{bmatrix} \tag{6}$$

3.2 Image Segmentation

The ideal output of the image segmentation operation is to group those pixels in the image which share certain visual characteristics that are perceptually meaningful to human observers. This is a difficult problem as humans use all sorts of tricks to perform this task. However, this task is very important as good segmentation can help with recognition, tracking, image database retrieval and image compression [19]. In this subsection we first present an overview about the various image

segmentation concepts and then we provide some of the most important segmentation tools that are used in our thesis.

3.2.1 Image Segmentation Overview

As explained in section 1.1, the segmentation step is one of the early stages of our project, thus the accuracy of this step plays a crucial role in the overall performance of our system. Generally, four segmentation techniques are introduced in the literature, including color segmentation, size and shape segmentation, and finally, texture segmentation [20], [21]. Since we have mostly focused on texture segmentation methods in this thesis and readers may be familiar with other segmentation schemes, we only introduce texture segmentation in the following subsection. It is clear that all of the mentioned segmentation methods and their corresponding tools are engaged in our food recognition system as will be described in subsequent chapters.

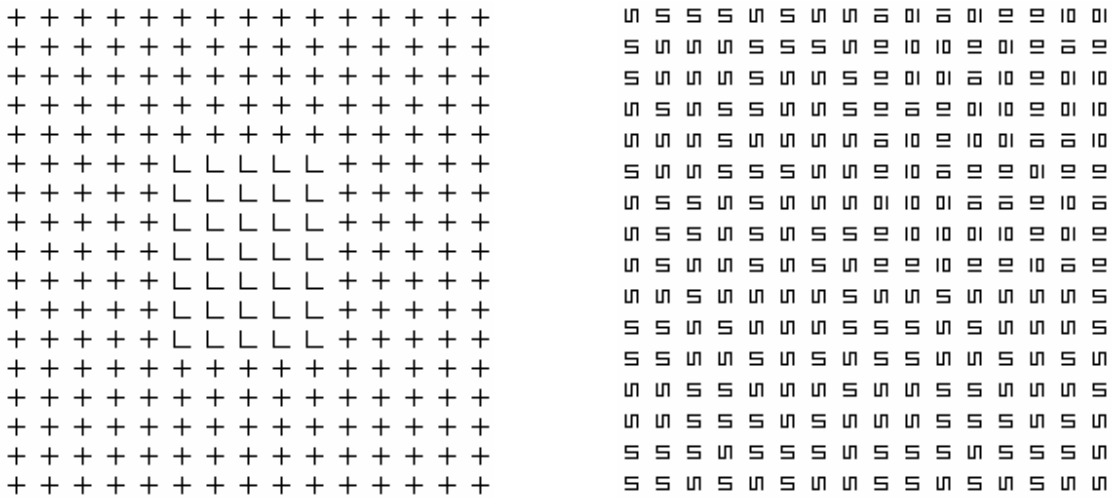
3.2.1.1 Texture Segmentation

Color and texture are fundamental characteristics of natural images, and play an important role in visual perception. Although color has been used in identifying objects for many years, texture is another active topic in segmentation. Since the 1950s, texture segmentation has been used in machine intelligence and pattern analysis which try to discriminate different patterns of images by extracting the dependency of intensity between pixels and their neighboring pixels [22], or by obtaining the variance of intensity across pixels [23]. In the recent research into food recognition areas, different features of color and texture are combined together in order to measure food nutrition more accurately [24].

Defining what a texture is or how to differentiate between two textures is a complex task. A possible reason for this difficulty is that humans are not consciously aware of what their eyes or

minds do when they identify a texture; the visual system seems to do it automatically. This automatic feature of the visual system is called pre-attentive vision, and this is where texture segmentation occurs [25].

In the early days of the perception research field, it was thought that the human visual system is capable of computing the Fourier Transform of an image and identifying textures based on their power range. However, this view has been mostly ignored, and the texton-based theory of texture perception is more widely accepted. The basics of texton theory are described by [26], in that textons are the central building blocks of texture perception. Textons are micro-structures that appear in an image and allow the visual system to see textures. For example, the ends of lines are textons. The authors of [26] believe that the visual system perceives differences in average texton density as differences in texture, but the relative positions of the textons to one another are ignored. For example, Figure 3-3 (a) is composed of + and L symbols whose line lengths are all equal. The + symbol contains a line crossing texton and four terminator textons, whereas the L symbol contains only two terminator textons. Since the symbols have different texton densities, they are seen as separate textures. However, Figure 3-3 (b) is composed of I0 and S shaped symbols rotated in various ways. Since both symbols have two terminator textons, they are of equal texton density and distinguishing them requires careful attention.



(a) (b)

Figure 3-3 (a) Image composed of L and + symbols is discriminable. (b) Image composed of S-I0 symbols is not discriminable without focal attention.

The objective of texture segmentation is to identify different uniform textural regions in an image. A texture segmentation example is shown in Figure 3-4. The image on the left is the original image consisting of three uniform textural regions. The image on the right presents an "ideal" segmentation result. In other words, the segmentation process produces an image of the same size as the original textural image, in which different colors represent different uniform textural regions in the original image.

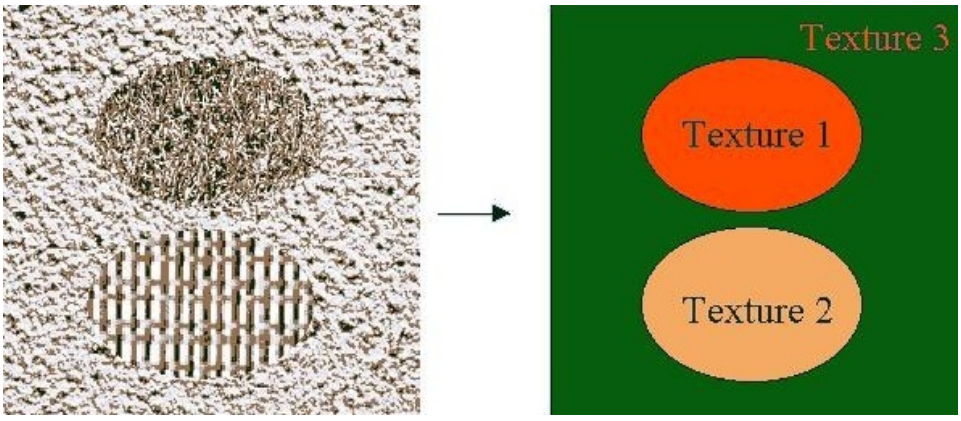


Figure 3-4 Texture segmentation. Left: original image, right: ideal segmentation result

There are several approaches for texture segmentation. Commonly, the segmentation process consists of two major steps, namely feature vector extraction and feature vector clustering. In feature vector extraction, several textural features are extracted from localized regions in the image. All features extracted from the same localized region comprise a single feature vector. In feature vector clustering, the feature vectors extracted from all localized regions are grouped into several groups. Ideally, all vectors belonging to the same group should correspond to visually similar local texture regions. Later in this section we will introduce the Gabor filter which has been used by numerous researchers in texture analysis problems [27] [28].

3.2.2 Image Segmentation Tools

In this subsection we will introduce a range of image segmentation tools which can be used in the various segmentation schemes introduced in subsection 3.2.1.

3.2.2.1 Threshold Techniques

Threshold techniques, which make decisions based on local pixel information, are effective when the intensity levels of objects fall outside the range of levels in the background. One of the threshold-based segmentation methods is Binary Large Objects (Blob) which is used for thumb measurement, as explained in section 4.2.5.

3.2.2.2 Image Filtering

In general, filters accept an input, such as a noisy image, process it and then convert it into an output, such as a de-noised image. In other words, a filter is a medium that allows only certain components of the input to pass through. In image processing, filters are used in several applications including edge detection, smoothing, and others.

In image segmentation, filters are usually represented by small matrices or masks. The masking operation is performed by continuously sliding the mask on the image, until the center of the mask travels over all image pixels. At each mask location, image pixels are multiplied with corresponding mask values, and the results of all multiplications are added to produce the final result. The result is then assigned as a pixel value in a new, masked image.

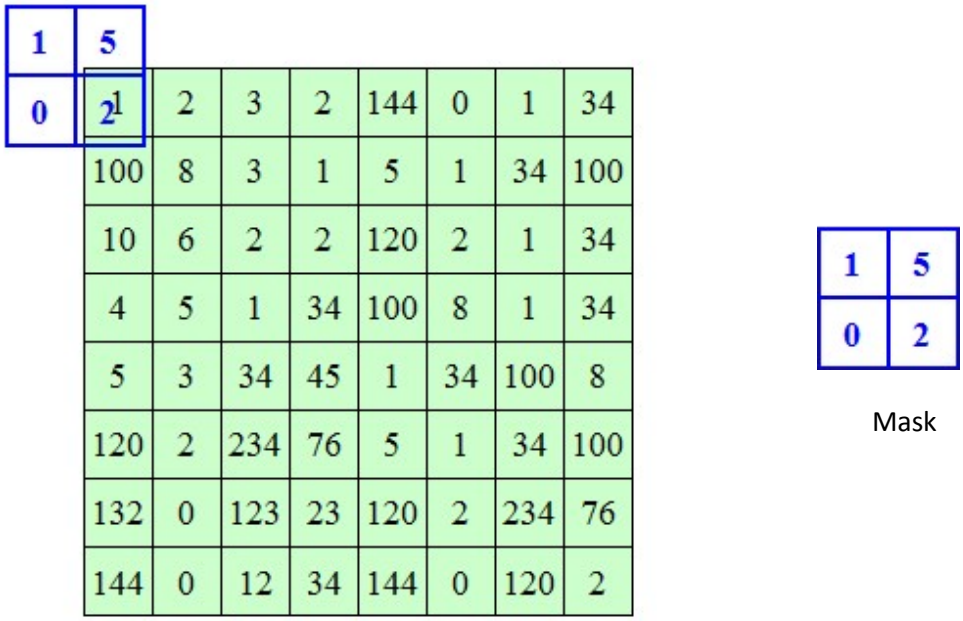


Figure 3-5 How to apply a mask to the image pixels (left: image pixels, right: mask).

For Figure 3-5, the masking operation works as follows. First, the center of the mask is placed at the leftmost upper corner of the image – let us define the mask size as $M_1 \times M_2$, where M_1 is the mask height and M_2 is the mask width. If both the mask width and height are odd, the mask center is at the array location ((M_1 − 1) / 2, (M_2 − 1) / 2).

Different types of filters can be constructed using two-dimensional functions, such as the Gaussian envelope function shown in formula (7).

$$G(x,y) = \exp\frac{1}{2\pi\sigma^2}\left\{-\frac{x^2+y^2}{2\sigma^2}\right\}, \tag{7}$$

where σ^2 is the variance of the Gaussian function. The constant $\frac{1}{2\pi\sigma^2}$ is used for normalization purposes.

3.2.2.3 Edge Detection

Edge detection is a fundamental tool in image processing, machine vision and computer vision, particularly in the areas of feature detection and feature extraction. This method aims to identify portions of a digital image where the brightness is changed sharply. Initially proposed by Roberts [29], edge detection can be performed using a double convolution of the image, with two kernels. To support his theory, Roberts proposed to process the image with a local differential operator. In this way a new image will be produced, whereas the edges are highlighted as much as possible. The final image will look like a simple line drawing, following all the possible contours of the objects present in the scene. Roberts' goals were to produce sharp edges, reduce the noise inside the image produced by the background, and finally achieve an image representation of the original picture close to the perception of the human eye. He used the functions in equations (8) and (9) to process the images and generate the differential results.

$$y_{i,j} = \sqrt{x_{i,j}} \tag{8}$$

$$z_{i,j} = \sqrt{\left(y_{i,j} - y_{i+1,j+1}\right)^2 \left(y_{i+1,j} - y_{i,j+1}\right)^2}, \tag{9}$$

where $x_{i,j}$ is the initial intensity value from the original image. Here, $z_{i,j}$ stores the result of the applied formulas. The sub-index i and j are the corresponding coordinates for the two dimensional matrix that represents both the input and the output images.

The same results can be obtained by convolving the image, using $\begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix}$ and $\begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}$ kernels.

A Gaussian operator is proposed in [30] which performs better than the abovementioned convolving scheme. In the following subsection we will explain this method in detail.

3.2.2.3.1 Convolution with Gaussian Operator

In order to detect the edges more precisely, Canny [30] proposed to find the image convolution using a Gaussian operator G_n extracted from the first derivative of a two dimensional Gaussian distribution. This method is similar to the kernel-based edge detection scheme [29], but it appears to be less expensive in computational time, and reduces background noise.

The Gaussian operator in any direction is given by formula (10).

$$G = \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{10}$$

Therefore:

$$G_n = \frac{\partial G}{\partial n} = n \,.\nabla G \tag{11}$$

Because the direction is not known, it is important to estimate the direction. To do so, a smooth gradient operation can be engaged as shown in (12).

$$n = \frac{\nabla(G\ conv\ Image)}{|\nabla(G\ conv\ Image)|} \tag{12}$$

By definition, the edges are local maxima in a direction of n. In order to obtain this local maximum, formula (13) is used.

$$\frac{\partial}{\partial n} G_n\ conv\ Image = 0 \tag{13}$$

And by substitution of G_n in formula (13), we can define a new combination as shown in (14).

$$\frac{\partial^2}{\partial n^2} G\ conv\ Image = 0 \tag{14}$$

And finally, the magnitude of the gradient will be given by formula (15).

$$|G_n \; conv \; Image| = |\nabla(G \; conv \; Image)| \tag{15}$$

3.2.2.4 Gabor Filter

The Gabor function, which is mainly used for texture detection, can be defined as shown in formula (16).

$$G(x,y) = \frac{1}{2\pi\sigma_x\sigma_y}\exp\left\{-\frac{x^2}{2\sigma_x^2}-\frac{y^2}{2\sigma_y^2}\right\}\cos(2\pi f_0+\varphi) \tag{16}$$

The Gabor filter has some directional possessions, as an example when $\sigma_x > \sigma_y$ the filter will look to be extended to the horizontal axis. The filter works as a band-pass filter by imagining the horizontal axis with a central frequency of f_0. Similarly, by imagining the vertical axis, it works as a low-pass filter.

If the Gabor filter has an orientation of θ, the following transformations can be done:

$$x' = xcos\theta + ysin\theta \tag{17}$$

$$y' = -xsin\theta + ycos\theta \tag{18}$$

Some Gabor filter examples for different σ_x, σ_y and f_0 parameters are shown in Figure 3-6.

The gray-level intensity of the surface plots is related to the value of the filter mask at the particular mask location (gray is 0, low and high intensities correspond to negative and positive mask values, respectively).

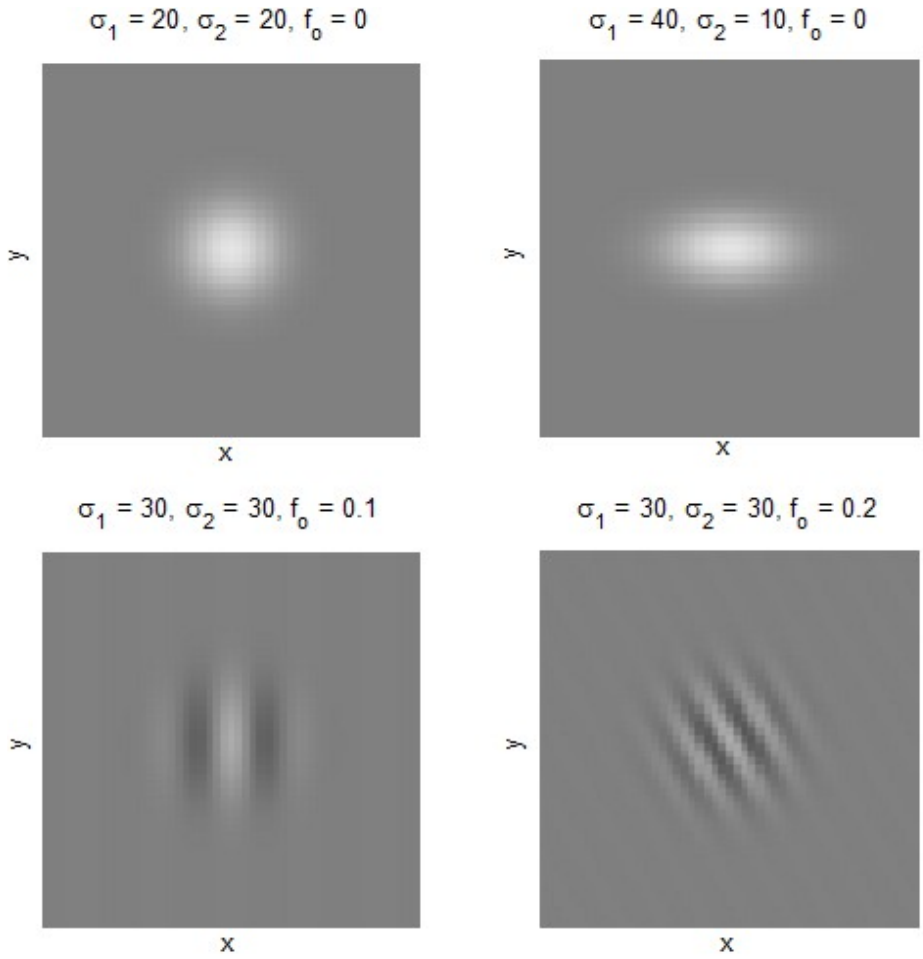


Figure 3-6 Examples of applying Gabor filter with various parameters.

3.2.2.5 K-mean Clustering

The K-means algorithm is used for clustering data points or vectors into a number of clusters.

In other words, it categorizes the pixels into a number of clusters, where each cluster represents a specific texture. For example, assuming a total of R vectors, $x_r = 1,2, \dots R$, for an image. The goal of the K-means algorithm is to group or cluster the R vectors into K groups. The algorithm of K-means is as follows:

1. Initialize K-cluster centroids (C_k) randomly.

A total of K vectors, $C_k = 1,2,3 \dots, k$ are chosen randomly, $x_r = x_{rL}, L = 1,2, \dots L$.

The vectors are called centroids. Each centroid is the typical vector of its own group.

2. Assign each sample to the nearest centroid.

Each vector x_r is assigned to one of the K groups. A usually used distance measure is the Euclidean distance which is shown in (19).

$$E^2\{x_r, C_k\} = (x_{r1} - c_{k1})^2 + (x_{r2} - c_{k2})^2 + \cdots + (x_{rL} - c_{kL})^2 \quad (19)$$

3. Calculate centroids (means) of K-clusters.
4. If centroids are unchanged, done. Otherwise, go to step 2.

3.2.2.6 Active Contour Extraction

An active contour, also known as a snake, is a curve defined by the specific characteristics of the object of interest inside the image. This curve will adapt itself following an energy minimization function [31]. For the energy minimization function, we need to understand the contour as a controlled spline under the manipulation of image and external forces. The function is shown in formula (20).

$$E^*_{snake} = \int_0^1 E_{snake}\big(v(s)\big)ds = \int_0^1 E_{int}\big(v(s)\big) + E_{image}\big(v(s)\big) + E_{con}(v(s))\, ds \quad (20)$$

The first integral of the equation is defined by the parameterization of the initial snake, given by $v(s) = (x(s), y(s))$. In the E_{int} section, the internal image energy spline is considered; meanwhile, E_{image} brings into the equation the energy force related to the image itself. Finally, the E_{con} helps us to consider the external forces that will also affect the active contour.

Different steps of a contour calculation are shown in Figure 3-7. The processed image is from a lung captured by X-ray. Part (a) shows the original image, while parts (b)-(e), represent how the active

contour is modified over the image in every step of the process. Part (f) is the combination of the original image (part (a)) and the final active contour (part (e)).

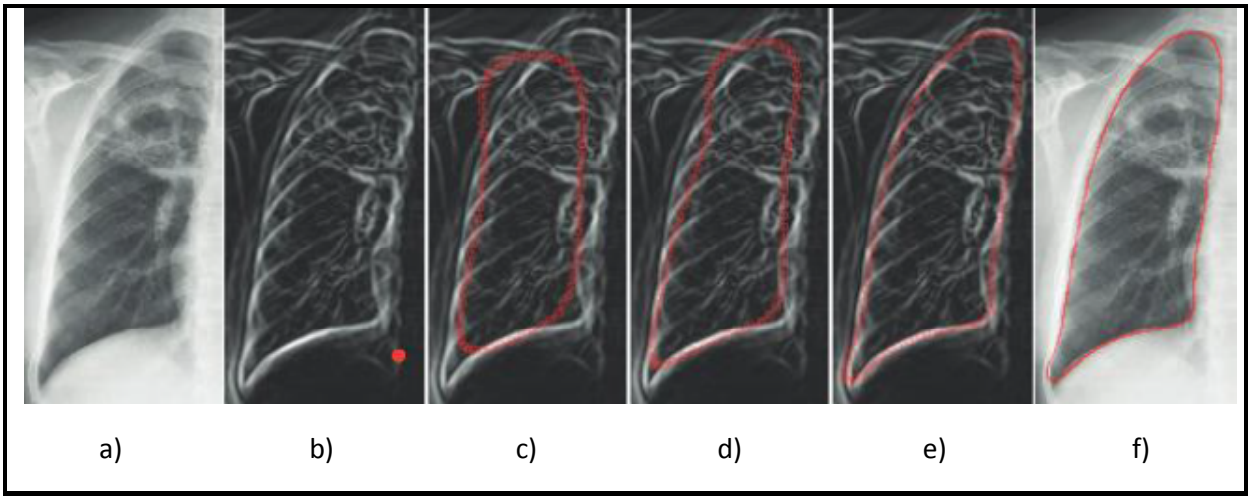
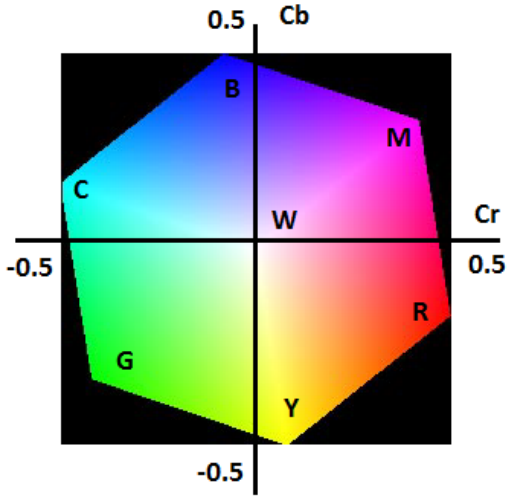


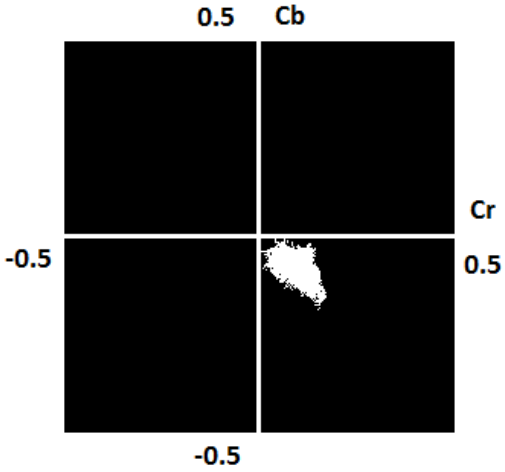
Figure 3-7 Images with an active contour example [32].

3.2.2.7 Skin Recognition

Based on skin and face detection investigations [33] [34], and physical experiments, it is known that one of the most common ways of perceiving colors, is using the Hue, Saturation and Intensity characteristics of the image. Human skin falls into a very specific category of colors, which make it unique to some extent. Parts (a) and (b) of Figure 3-8, show the distribution of the common RGB colors and skin colors over the Chrominance Cb and Cr components. Based on the fact that Cb and Cr consider hue and saturation, skin tones are relatively consistent in the chrominance plane.



a) RGB distribution over CbCr



b) Human skin Distribution over CbCr

Figure 3-8 Images of CbCr colors and skin location.

Once the area of focus has been defined, a characterizer must be applied in order to define the pixels corresponding to a possible skin tone and the pixels that do not correspond to a skin tone. To do so, the Bayesian decision rule for minimum cost can be applied, as shown in formulas (21)-(24).

$$R_0(X) = C_{00}p(w_0|X) + C_{10}p(w_1|X) \quad (21)$$

$$R_1(X) = C_{01}p(w_0|X) + C_{11}p(w_1|X) \quad (22)$$

$$R_0(X) < R_1(X) \rightarrow X \in w_0 \quad (23)$$

$$R_0(X) > R_1(X) \rightarrow X \in w_1 \quad (24)$$

For these formulas, the term w_0 corresponds to the pixels with no skin color, and the term w_1 corresponds to the pixels with skin color. The term $p(w_i|X)$ corresponds to the probability of being in a specific class, based on the sample X. The terms C_{00} and C_{11}, are related to the cost coefficient of the correct classification, and C_{01} and C_{10} are the costs of misclassifications. These formulas will help us to classify the pixels, following a simple rule of minimum cost decision making, as shown in equations (25) and (26).

$$C_{10}p(w_1|X) < C_{01}p(w_0|X) \rightarrow X \in w_0 \quad (25)$$

$$C_{10}p(w_1|X) > C_{01}p(w_0|X) \rightarrow X \in w_1 \quad (26)$$

This kind of analysis becomes a practical problem with a possible solution, thanks to the characteristics of the color space used to analyze the information. For calculating the size of a finger in our project we used this method, which we will be discussed in the following chapters.

3.3 Feature Extraction

An essential step in solving any object classification task is to represent the visual information of the object. This is commonly known as feature extraction. Features need to be robust to image perturbations, such as viewpoint and illumination changes.

Given the labeled pixels of an image after segmentation, we need to extract visual characteristics from each region (food item) to help train the classifier to identify each food item. Therefore, proper selection of features is the key to correct classification.

Certain criteria are necessary when selecting the features: the features should contain enough information to uniquely describe each food item yet not be domain-specific; they should be easy to compute and relate well with the human visual system. Based on these criteria, four types of features can be extracted for each segmented food region, which are: color, size, shape and texture.

Color image quantization (clustering) is a process that reduces the number of distinct colors used in an image, usually with the intention that the new image should be as visually similar as possible to the original image. Figure 3-9 shows colorful images reduced to four color clustering.

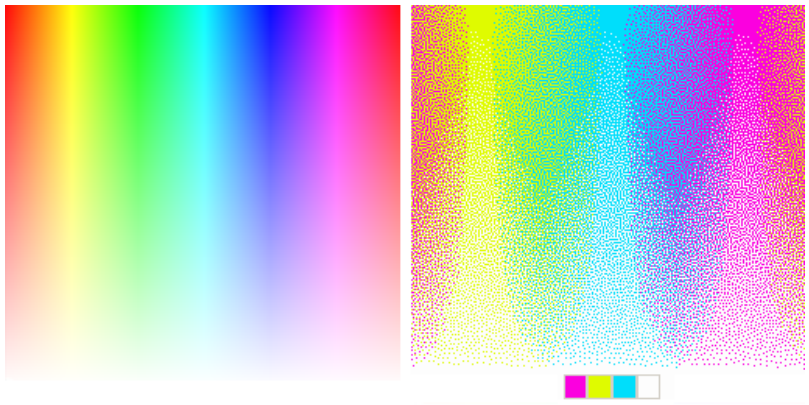


Figure 3-9 Colorful images reduced to four colors using spatial color quantization.

For color feature extraction, color mean shift segmentation followed by color-clustering can be applied to the image. For texture features, Gabor filters can be used to measure local texture properties in the frequency domain.

Shape feature learning methods extract individual shape features from images. Shape features are the features of objects constructed by a set of elements like points, lines and curves. These features can be corner features or edge features which can be detected by feature detection methods. Edge detection and counter definition can be engaged to find the shape of food portions.

Size based features consist of the ratio between the area of the detected object to the area of the original object [35]. Area is represented by the number of pixels in the respective components. More details about the feature extraction step will be provided in chapter 4.

3.4 Classification

Once the food items are segmented and their features are extracted, the next step is to identify the food items using statistical pattern recognition techniques [36], [37]. Classifiers are used in a variety of pattern recognition and machine learning applications ranging from automatic speech recognition to the characterization of faces.

The recognition process often consists of two steps. In the training phase, a model is learned using training data. The model is tested using unseen data to assess the model's accuracy in the testing phase. Through training, the classifier learns how to map the features into various categories or labels. However, the training is not perfect and the classifier will make mistakes in actual operation by assigning the wrong label to an observed feature vector.

Training the classifier has two goals. One goal is to define how it will assign labels to observed feature vectors and the other is to estimate its error performance or its classification accuracy.

Support vector machines are a set of manageable learning methods used for classification and regression [38]. They belong to a family of global linear classifiers. In other words, the support vector machine is a classification and regression device that uses machine learning theory to maximize predictive accuracy to automatically fit data into the best category. Support vector machines can be defined as systems which use the hypothesis space of linear functions in a high dimensional feature space.

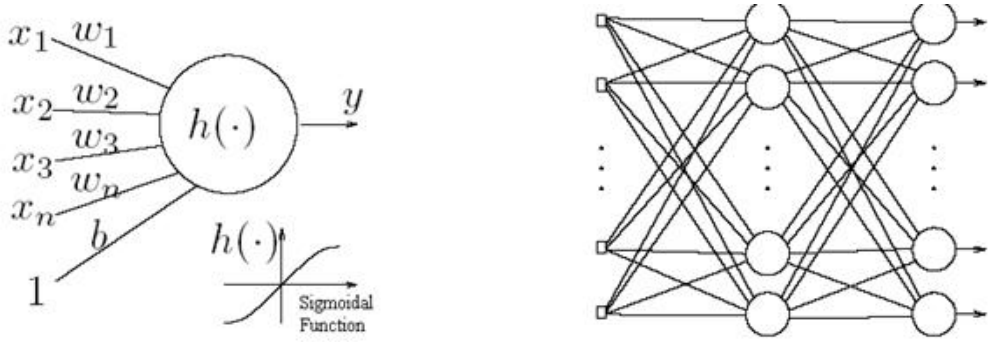
SVM gives accurate results compared with neural network methods in the same task, e.g. the handwriting recognition task in [39]. It is also being used for many applications, such as face analysis and speech recognition, especially for pattern classification and regression based applications [38].

SVMs were developed to solve the classification problem, but recently they have also been extended to solve regression problems [40].

3.4.1 Why the Support Vector Machine?

The neural network technique is one of the recognition methods used for supervised and unsupervised learning applications and it has had good results. One neural network method is the Multilayer Perceptron (MLP) algorithm. MLP uses feed forward and recurrent networks. This

algorithm includes universal approximation of continuous nonlinear functions, and includes learning with input-output patterns [41], [42]. Figure 3-10, shows the simple neural network and multilayer perception's behavior.



(a) (b)

Figure 3-10 a) Simple neural network, b) multilayer perceptron [41].

But there are some problems with MLP which encourage people to find and use other methods.

First, having many local minima and difficulties finding the number of neurons that might be needed for a task persuades us to test other learning methods. In addition, even if the used neural network solutions have a tendency to converge, this will not be a unique solution. As we know in neural network methods, we have to try and test to find an optimum solution and this may take a long time [43], [44].

As known, there are many linear classifiers (hyperplanes) that separate the data. If we use a linear hyperplane to classify data, we might end up close to one set of datasets which is a local minimum but not a global minimum. Therefore we see the concept of a maximum margin classifier as an obvious solution. Figure 3-11 gives the maximum margin classifier example which provides a solution to the abovementioned problem [45]. In the following section, this will be further explained.

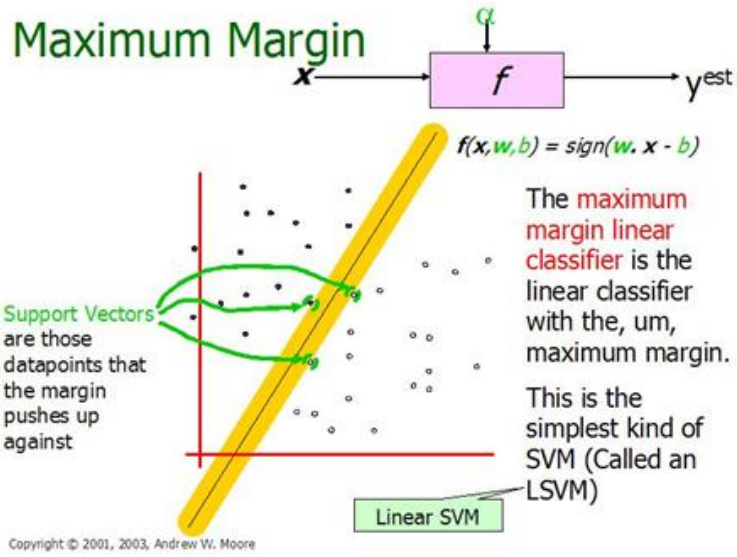


Figure 3-11 Linear SVM [46].

In a linear SVM, the maximum margin is used. A question that may arise here is why maximum margin? One reason is that even if we've made a small error in the location of the boundary, this gives us minimum chance for causing a misclassification. The other advantage would be avoiding local minima and ultimately getting a better classification.

Now let us express the SVM mathematically. The goals of SVM are separating the data with a hyperplane and extending this to non-linear boundaries using the kernel trick [45], [46]. For calculating the SVM, the goal is to correctly classify all of the data. For mathematical calculations we have,

$$x_i.w + b \geq +1 \qquad for\ y_i = +1 \tag{27}$$

$$x_i.w + b \leq -1 \qquad for\ y_i = -1 \tag{28}$$

These can be combined into one set of differences:

$$y_i(x_i.w + b) - 1 \geq 0 \qquad \forall i \tag{29}$$

In this equation x is a vector point and w is weight and is also a vector. So to separate, the data, (27) should always be greater than zero. Among all possible hyperplanes, SVM selects the one

where the distance of the hyperplane is as large as possible. If the training data are good and every test vector is located in radius r from the training vector, it will be chosen as a default. Now, if the chosen hyperplane is located at the farthest possible point from the data then this desired hyperplane – which maximizes the margin – also bisects the lines between the closest points on the convex hull of the two datasets. Thus we have (27), (28), (29). The implementation of this part is shown in Figure 3-12.

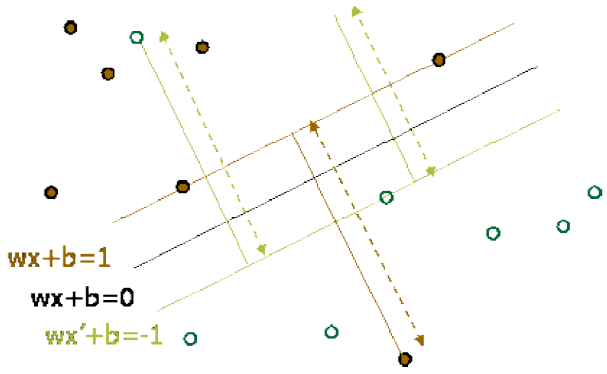


Figure 3-12 Representation of hyperplanes [46].

The distance of the closest point on the hyperplane to the origin can be found by maximizing the x, as x is on the hyperplane. Similarly for the other side points we have a similar algorithm. Thus , by solving and subtracting the two distances we get the summed distance from the separating hyperplane to the nearest points.

$$Maximum\ Margin\ =\ M\ =\ 2\ /\ ||w||.$$

A unique solution exists for maximizing this quantity using Lagrange's theorem [13], [47]:

$$L_D = \sum_{i=1}^{l} \propto_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \propto_i \propto_j y_i y_j x_i . x_j \qquad (30)$$

Under these constraints:

$$\propto_i \geq 0 \ \forall_i, \qquad \sum_{i=1}^{l} \propto_i y_i = 0, \tag{31}$$

where, $\propto_1, \ldots, \propto_n$ are the Lagrange multipliers.

A nonlinear SVM is used in place of the linear SVM where the training vectors are not linearly separable. With a nonlinear SVM, the training vectors are separated by a nonlinear boundary. The nonlinear decision boundaries are determined by mapping the training vectors to some other Euclidian space.

A hyperplane is then used to separate the mapped training vectors. To avoid complexity increase due to mapping the training vectors to a high dimension, the mapping of $\Phi(\mathrm{x_i}).\Phi(\mathrm{x_j})$ is replaced by a Kernel Function as shown in formula (32).

$$\mathrm{k}(\mathrm{x_i}, \mathrm{x_j}) = \Phi(\mathrm{x_i}).\Phi(\mathrm{x_j}) \tag{32}$$

The Lagrangian of (32) is then redefined for the nonlinear case by substituting $\mathrm{k}(\mathrm{x_i}, \mathrm{x_j})$ in the place of $(\mathrm{x_i}, \mathrm{x_j})$. This produces the modified maximization problem formulated in (33).

$$\mathrm{L_D} = \sum_{\mathrm{i=1}}^{\mathrm{l}} \propto_{\mathrm{i}} - \frac{1}{2} \sum_{\mathrm{i=1}}^{\mathrm{l}} \sum_{\mathrm{j=1}}^{\mathrm{l}} \propto_{\mathrm{i}} \propto_{\mathrm{j}} \mathrm{y_i y_j k}(\mathrm{x_i}.\mathrm{x_j}) \tag{33}$$

The SVM classifier generates its training model using the LIBSVM [47], a library for support vector machines that supports multi-class classification.

Furthermore, $\mathrm{k}(\mathrm{x_i}, \mathrm{x_j}) = \Phi(\mathrm{x_i}).\Phi(\mathrm{x_j})$ is called the kernel function. The SVM used one of these four kernels which are shown in (34), (35), (36) and (37).

- Linear: $\mathrm{k}(\mathrm{x_i}.\mathrm{x_j}) = x_i^T x_j$ (34)
- Radial Basis Function (RBF): $k(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \gamma > 0$ (35)
- Polynomial: $\mathrm{k}(\mathrm{x_i}.\mathrm{x_j}) = (\gamma x_i^T x_j + r)^d$ (36)
- Sigmoid: $\mathrm{k}(\mathrm{x_i}.\mathrm{x_j}) = tanh\,(\gamma x_i^T x_j + r)$ (37)

In the above equations γ, r and d are kernel parameters.

For this project we propose the following procedure to use SVM in food classification:

- Transform data to the format of an SVM package
- Prepare data for simple scaling
- Consider the RBF kernel $k(x_i,x_j) = \exp\left(-\gamma\|x_i-x_j\|^2\right)$
- Use cross-validation to find the best C and γ parameters
- Use the C and γ parameters to train the whole training set
- Test

We will describe some of these steps in detail in the following section.

3.4.2 Data Pre-processing

3.4.2.1 Feature Categorization

In SVM, all data are represented as a vector of real numbers. So for each feature we have to convert their value to numeric data. One way is to show them using a combination of 0, 1 numbers. For example, a three-category attribute such as red, green, blue can be represented as (0,0,1), (0,1,0), and (1,0,0).

3.4.2.2 Scaling

The important fact that can be reached by scaling is avoiding difficulties in numerical calculation.

Because kernel values usually depend on the central products of feature vectors, it is better to use scaling in the range of [-1; +1] or [0; 1]. Please note that we have to use the same method to scale both training and testing data.

3.4.2.3 RBF Kernel

The RBF kernel is a reasonable first choice. This kernel maps the samples into a higher dimensional space, nonlinearly. So unlike the linear kernel, it can handle a case when the relation between class labels and attributes is nonlinear.

The second reason is the number of hyper parameters which increase the complexity of model selection. The polynomial kernel has more hyper parameters than the RBF kernel. Finally, the RBF kernel has fewer numerical difficulties.

There are some situations where the RBF kernel is not suitable. For example, when the number of features is very large, one may just use the linear kernel.

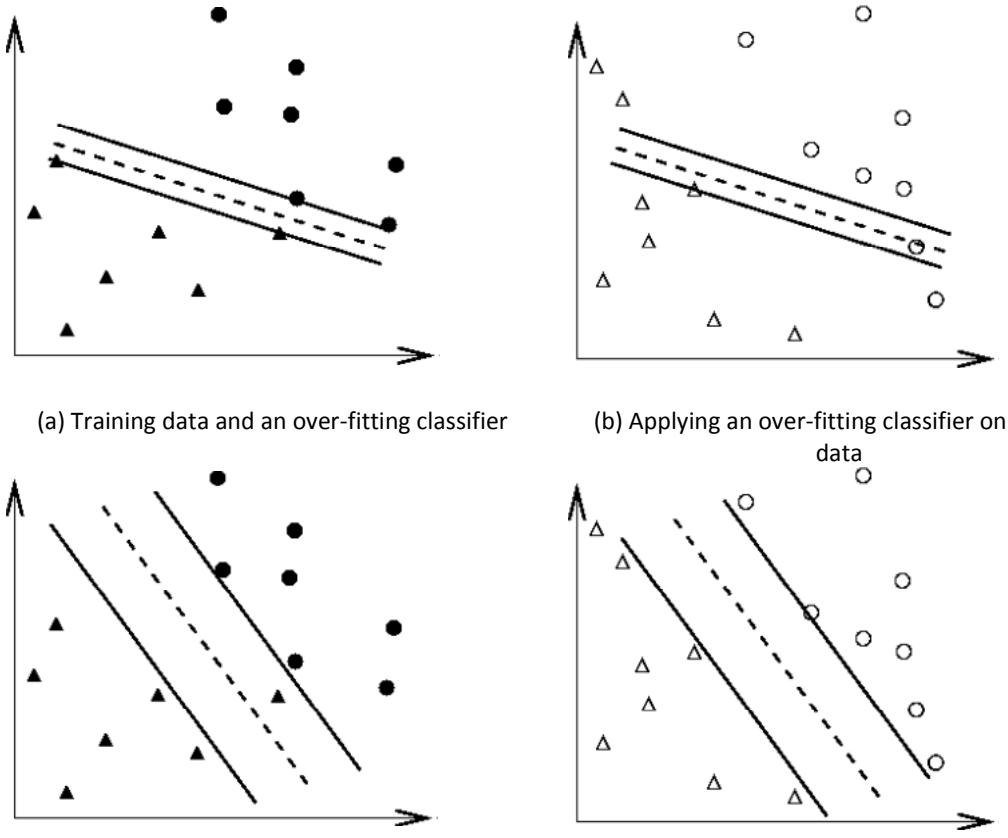
3.4.2.4 Cross-validation and Grid-search

There are two parameters for an RBF kernel: C and γ. The goal of this step is to find the best value for C and γ, so that the classifier can accurately predict unknown data (i.e., testing data).

A common strategy in this classifier is to separate the dataset into two parts of which one is considered unknown. The prediction accuracy obtained from the unknown set more precisely reflects the performance on classifying an independent dataset. An improved version of this procedure is known as cross-validation.

The advantage of using cross-validation is in preventing an over-fitting problem. Figure 3-13 represents a binary classification problem to illustrate this issue. Filled circles and triangles are the training data while hollow circles and triangles are the testing data.

The testing accuracy of the classifier in Figure 3-13 (a) and (b) is not good since it over-fits the training data. If we think of the training and testing data in Figure 3-13 (a) and (b) as the training and validation sets in cross-validation, the accuracy is not good. On the other hand, the classifier in (c) and (d) does not over-fit the training data and gives better cross-validation as well as testing accuracy. A good way to find C and γ is using “grid-search".



(a) Training data and an over-fitting classifier

(b) Applying an over-fitting classifier on testing data

(c) Training data and a better classifier

(d) Applying a better classifier on testing data

Figure 3-13: An over-fitting classifier and a better classifier (● and ▲: training data; ○ and △: testing data).

Chapter 4 The Proposed System

In this chapter we discuss our proposed system in more detail. In this regard, we explain various parts, one by one. A detailed block diagram of the proposed method is shown in Figure 4-1. As the figure shows, at the early stage, images are taken by the user with a mobile device followed by a pre-processing step. Then, at the segmentation step, each image will be analyzed to extract various segments of the food portion. It is known that without having a good image segmentation mechanism, it is not possible to process the image appropriately. Hence, we have worked on the segmentation step more in this project. To do so, we have used color segmentation, k-mean clustering, and texture segmentation tools. We will show how these steps lead to an accurate food separation scheme. For each detected food portion, a feature extraction process has to be performed. In this step, various food features including size, shape, color and texture will be extracted. The extracted features will be sent to the classification step where, using the SVM scheme, the food portion will be identified. Finally, by estimating the area of the food portion and using some nutritional tables, the caloric value of the food will be calculated.

In the following subsection we will explain each step of Figure 4-1 in more detail.

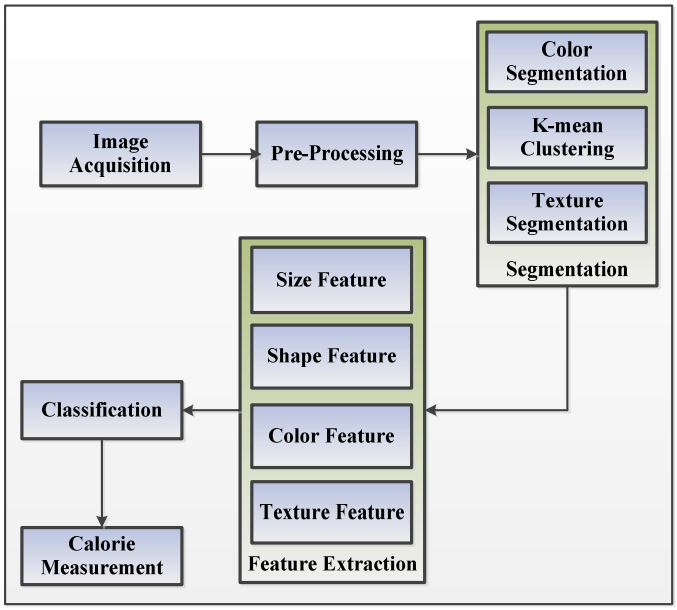


Figure 4-1 Image analysis system.

4.1 Pre-processing

First of all, in order to have accurate results for our segmentation, a simple transformation must be performed on the image to change the image size into standard format. To do so, the size of each image will be compared with standard size categorizes. If the image size is not compatible with any size category, some cropping or padding techniques will be applied to the image.

In this project we have defined one size category, i.e. 970×720 for simplicity. Larger images will be adjusted to this size, before performing any image processing technique.

4.2 Image Segmentation

In the following parts, we will explain the image segmentation tools which are adopted in our system.

4.2.1 K-mean Clustering

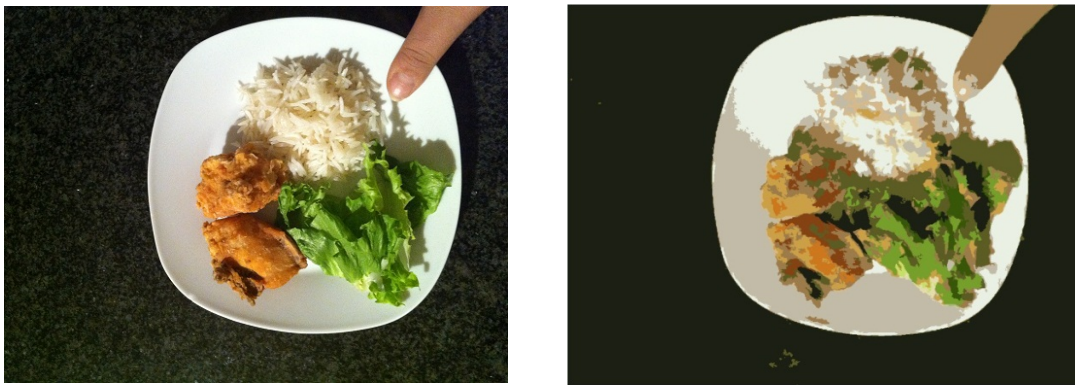
There are many clustering algorithms in the literature such as mean shift and k-means. The mean shift algorithm is a nonparametric clustering technique which does not require prior knowledge of the number of clusters, and does not constrain the shape of the clusters. The k-mean algorithm iteratively computes the mean of a set of clusters, until it converges to a stable set of cluster samples. In gray-scale images, areas are typically modeled as uniform intensity areas. Segmentation algorithms employ some form of Euclidean distance measure to determine pixel similarity either on a spatially local basis or on a global color basis. For color image processing, the clustering algorithms operate in complex multidimensional spaces. Because of the added complexity of needing three variables to represent color pixels, the issue of region segmentation in color images is not as well defined as for gray-scale images.

In the segmentation step of this project, we focus more on creating regions of similar color. This means that the choice of distance measure becomes very important since similarity depends very much on how distances between colors are being measured. In this case, all approaches found in the literature use some form of Euclidean distance to determine similarity between two color pixels.

In our project, we first smooth the RGB histogram using Gaussian edge detection, introduced in section 3.2.2.3, and then apply closing and expansion operators to segment the image. In other words, the hired k-mean clustering method is as follows:

1. Initialize centroids of K-clusters randomly.
2. Assign each sample to the nearest centroid.
3. Calculate centroids (means) of K-clusters.
4. If centroids are unchanged, done. Otherwise, go to step 2.

Figure 4-2 shows the result of this clustering scheme on a sample food image. As the figure shows, different colors are clustered appropriately.



a) Original image b) Image with color clustering

Figure 4-2: Color clustering images.

4.2.2 Texture Segmentation

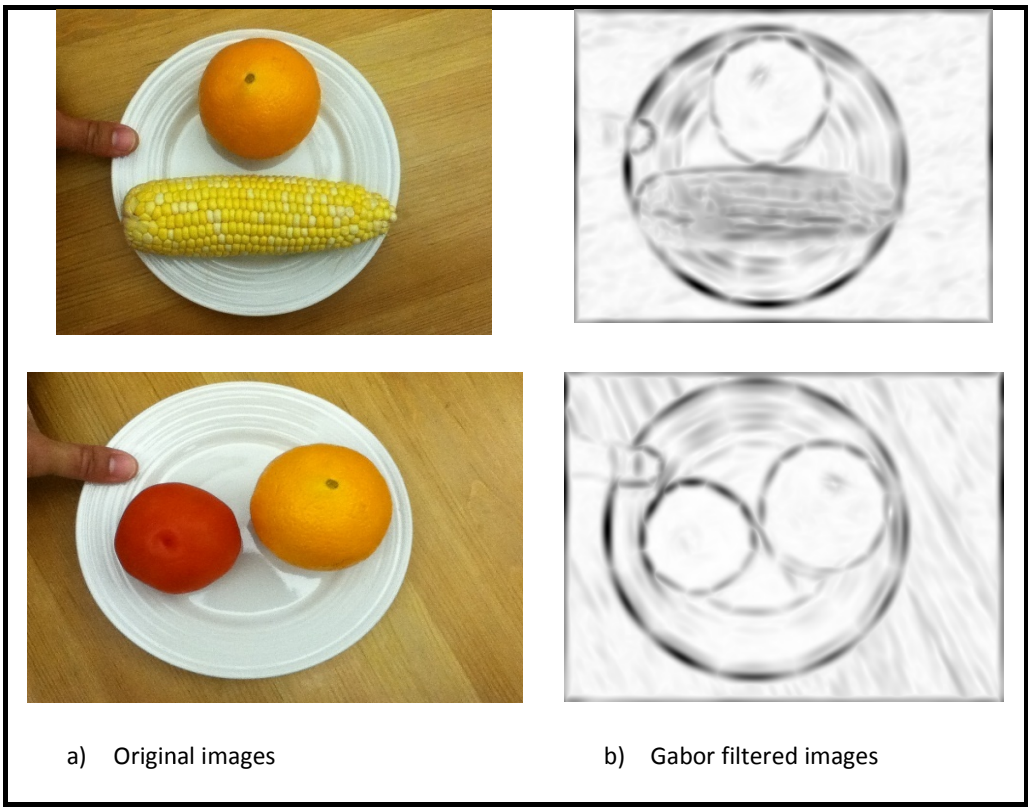
To obtain more accurate results in the segmentation stage, we also added texture segmentation to the method. For texture features, we used a Gabor filter to measure local texture properties in the frequency domain. As explained before, the Gabor filter describes properties related to the local power spectrum of a signal and has been used for texture analysis [48]. A Gabor impulse response in the spatial domain consists of a sinusoidal plane wave of some orientation and frequency modulated by a two-dimensional Gaussian envelope and is given by formula (38).

$$h(x,y) = \exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2}+\frac{y^2}{\sigma_y^2}\right)\right]\cos\left(2\pi Ux+\varphi\right) \quad (38)$$

In our work, we have used the Gabor filter-bank proposed in [49]. It is suitable for our use, where texture features are obtained by subjecting each image to a Gabor filtering operation in a window around each pixel and then estimating the mean and the standard deviation of the energy of the filtered image. A Gabor filter-bank consists of Gabor filters with Gaussians of several sizes modulated by sinusoidal plane waves of different orientations from the same Gabor-root filter as defined in (38).

In our implementation, a bank of Gabor filters with six different desired orientations and five wavelengths are applied to the image. Furthermore, we have included the spatial coordinates of the pixels as two additional features that we have in our segmentation to get an accurate result in this part. The outcome of each of these Gabor filters is a two-dimensional array, with the same size of input image. The sum of all elements in one such array is a number that represents the matching orientation and spatial frequency of the input image.

The results of applying a Gabor filter on food images are shown in Figure 4-3.



a) Original images b) Gabor filtered images

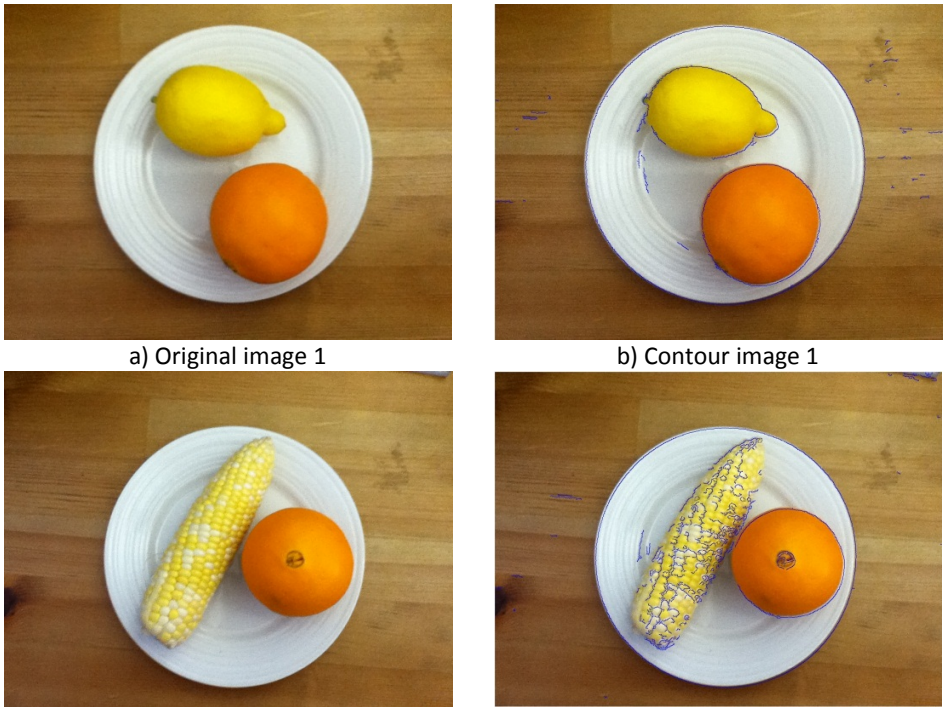
Figure 4-3 Gabor filter results.

4.2.2.1 Contour Description

At this point of the image segmentation and analysis, based on the characteristics demarked by k-mean color segmentation and also texture segmentation, contour definition will be extracted. The results of these image segmentations are a set of segments that collectively cover the entire image,

or a set of contours extracted from the image. Within each region, some of the pixels are similar with respect to a few characteristics, such as color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic. Now we just need to follow the active contours defined by the sharpness incremental procedure and the enhancement of the meaningful characteristics of the image. The extraction of those contours from the image will give us a complete set of pixels with its corresponding coordinates in the plane; and it will depict the simple shapes of the objects present inside the scene. Once the set of pixels has been defined, we can recreate the contours artificially, with strong colors such as red green or blue, just by following these pixels, and drawing lines between them.

Figure 4-4 shows two examples of the contour definition with blue colors, over regular images using the entire set under analysis.



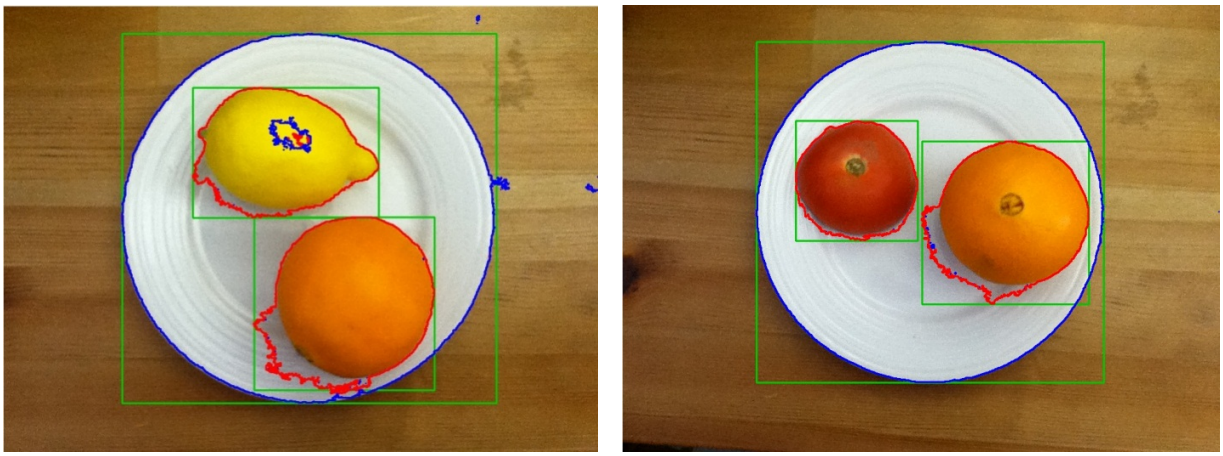
a) Original image 1 b) Contour image 1

c) Original image 2 d) Contour image 2

Figure 4-4 Contour definition images.

4.2.3 Area of Interest Explanation

In this step we need to extract the region of interest which we found from the segmentation part. By using our contour detection method we extracted each part from the background, so in our algorithm we can go through the detection step and extract it from the images. Figure 4-5 shows the result of this part.



a) Image 1 with contours and ROI defined

b) Image 2 with contours and ROI defined

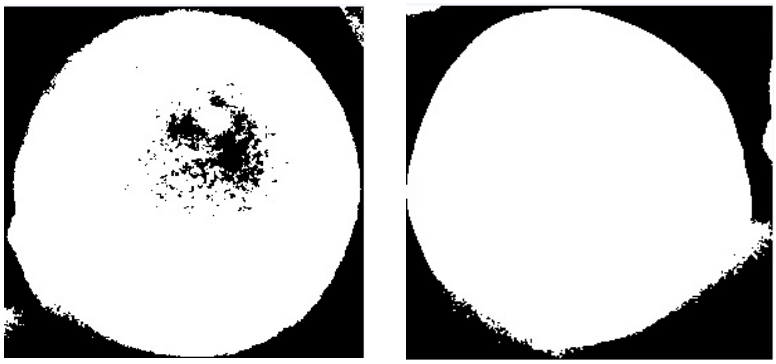
Figure 4-5: ROI definition images.

But in this level we need to calculate the size of each region of interest, as we want to calculate their calories. To do so, we use Blob, which is a collection of binary data stored as a single entity. "Blob" was initially used as a term for moving large amounts of data from one database to another without filters or error correction.

Now we want to transform ROI definition into a binary representation of the object. To do so, the pixels within the detected ROI positions will be changed to white and the rest of the image will be changed to black.

Figure 4-6 shows the two objects extracted and transformed; it is easy to understand at this point the reason why these types of images are called Binary Large Objects, due to the ratio in size

between the object and the image size. Now we can calculate the size of this area in pixels by counting the number of white pixels in each ROI.



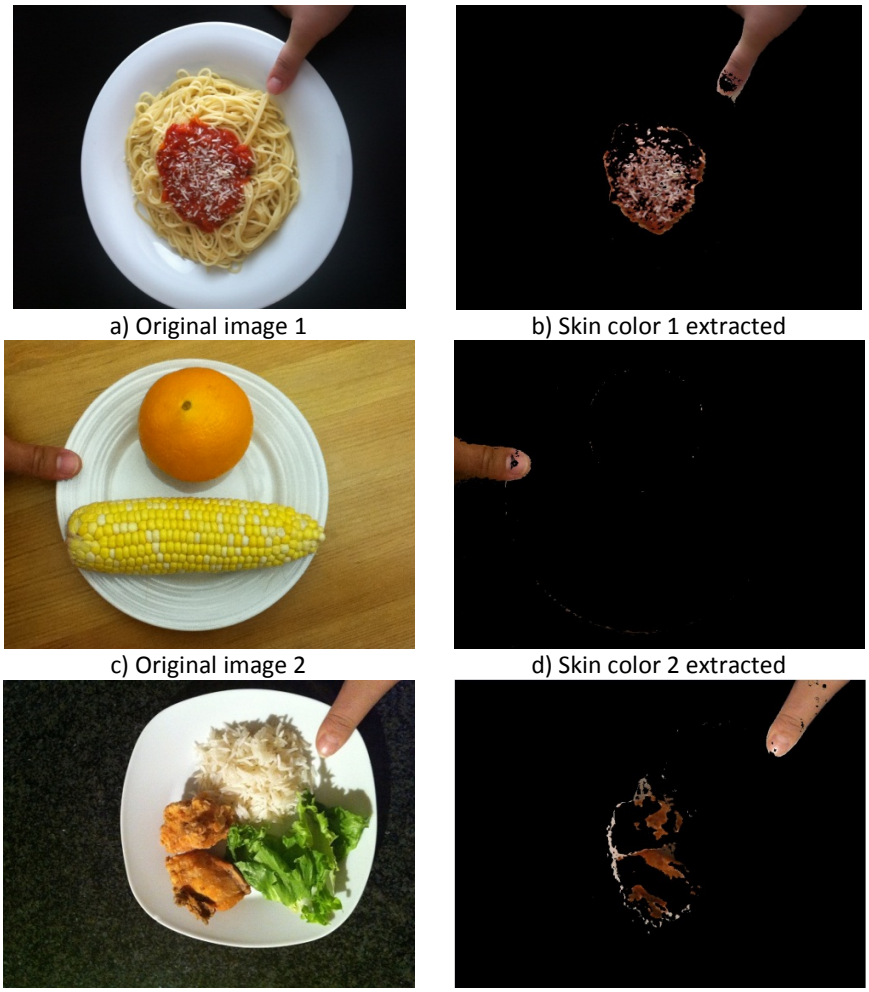
a) ROI 1 extracted b) ROI 2 extracted

Figure 4-6 Blob definition images.

4.2.4 Skin Color Definition

In order to obtain sections of the image corresponding to the patient's thumb, a special transformation is performed over the image to move from RGB color space to YCbCr color space. With these changes to the image, the analysis for the specific range of colors where the skin of a person can be located is applied to the images. The sections corresponding to pixels that fall into the category of skin color are kept in the image, the other pixels are removed, and black pixels replace them instead.

Figure 4-7 shows a set of original images (a, c, e) and extraction of the pixels corresponding to skin color (b, d, f).



a) Original image 1 b) Skin color 1 extracted

c) Original image 2 d) Skin color 2 extracted

e) Original image 3 f) Skin color 3 extracted

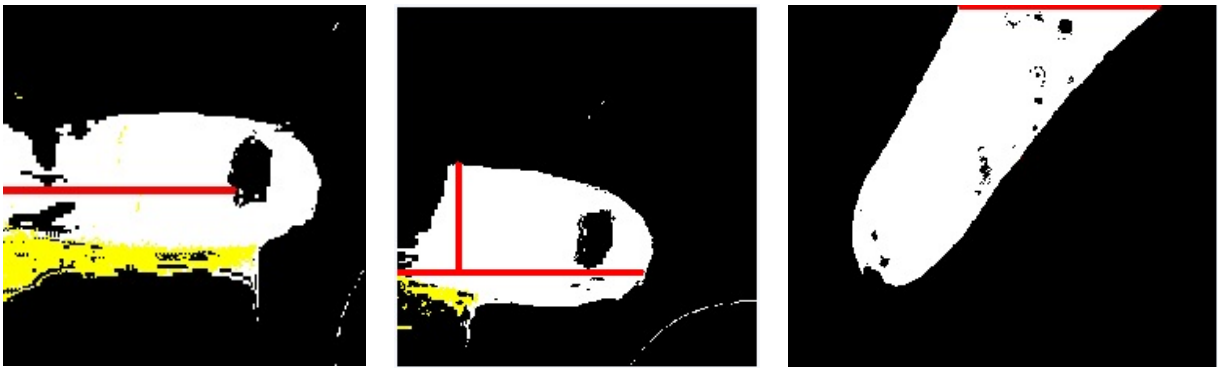
Figure 4-7: Images of skin color extraction.

4.2.5 Thumb Measurement[1]

Using the images with the human skin highlighted, the definition of the section of interest containing the thumb is extracted. The extracted area is then used to produce the binary large object corresponding to the thumb itself. Finally, when the Blob of the thumb is available, the dimensions of the thumb in pixels are extracted. Even when the Blob definition transforms the image into black and white pixels, we have been working on a three channel image, to be able to use special colors to signalize the dimensions of the thumb.

[1] This part was carried out by Gregorio Villalobos

Figure 4-8 shows the Blobs of the thumb from three different images; note that before the Blob extraction, the ROI of the image has been defined per image in order to focus the processing procedure only in a small area of the whole picture, to reduce processing overheads.



a) Thumb Blob 1 b) Thumb Blob 2 b) Thumb Blob 3

Figure 4-8: ROI and Blob of user's thumb images.

The data exposed from the image is in pixels, so with these data we cannot do the calculation part of the nutritional intake and calorie measurement. Hence, we have to convert the data to an actual result in order to use them for our size calculation and also for the last part which will be presented for the user as a result of our method in actual and real size dimension.

4.3 Feature Extraction and Classification

This section briefly describes the features used in our experiments. We have proposed a framework that uses four types of features, including color, texture, size and shape. For the color feature we used edge detection and color k-mean clustering. For the texture feature we used Gabor filtering. For the size and shape features we engaged the edge detection technique and counted the pixels inside the ROI area of the image.

The next step is classifying the extracted features in order to recognize each food portion. To do so, we used SVM which is one of the popular techniques used for data classification. A classification task usually involves training and testing data which consist of some data instances. Each instance in

the training set contains one class label and several features. The goal of SVM is to produce a model which predicts the target value of data instances in the testing set which are given only by their attributes.

To increase accuracy, after the SVM module has determined each food portion type, the system can optionally interact with the user to verify the kind of food portions. For instance, it can show a picture of the food to the user, annotated with what it believes are the portion types, such as chicken, meat, vegetable, etc. as described in [50]. The user can then confirm or change the food type. This changes the system from an automatic one into a semi-automatic one; however, it will increase the accuracy of the system.

In our model, we used the RBF kernel, which maps samples into a higher dimensional space in a non-linear manner. Unlike linear kernels, the RBF kernel is well suited for the cases in which the relation between class labels and attributes is nonlinear.

In our proposed method, the feature vectors of SVM contain five texture features, ten color features, three shape features, and six size features. The feature vectors of each food item, extracted during the segmentation phase, will be used as the training vectors for SVM.

4.4 Measurement Unit[2]

4.4.1 Calorie Definition and Nutritional Tables

The calorie is a typical measuring unit which is defined as the amount of heat energy needed to raise the temperature of one gram of water by one degree [3]. In other words, this unit is used to measure the overall amount of energy in any food portion that consists of the main food components which are Carbohydrate, Protein and Fat. Beside gram units, calorie units are adopted

[2] In this part I received assistance from Rana Almaghrebi

in developing nutritional fact tables. Each person should daily intake a certain amount of calories. If this amount is increased, it will lead to weight gain.

Table 1 illustrates the range of calorie units for various food. The proposed system in this thesis relies on the already-established tables as a reference to estimate the number of calories from any selected food photo.

Table 1 Nutrient table for some sample food.

Food Name	**Measure**	**Weight**	**Energy**	**Protein**	**Carb**	**Fat**
		(g)	***(kcal)***	***(g)***	***(g)***	***(g)***
Apple with skin	1	138	72	N/A	19	N/A
Potato, boiled, no skin	1	135	116	2	27	N/A
Chicken, ground, lean	75	75	153	16	0	9
Orange	1	131	62	1	15	N/A
Tomato, raw	1	123	22	1	5	N/A
Bread, white, commercial	1	35	93	3	18	1

4.4.2 Shape Recognition as Part of Image Processing

Shape recognition is a sub-area of image processing focused on the definition of different types of characteristics attained from every object present inside an image. The most common characteristics obtained are area, perimeter, size and Euler number (E). E is defined by the number of connected components (C), and the number of object holes (H) given as E=C-H. Geometric attributes are closely related to a standard geometrical form such as circle, square or triangle.

In the following section, we describe the basic concepts that we depend upon to build our system.

4.4.3 Obtaining Volume by Utilizing Area Size

The proposed Food Recognition System (FRS) is based on a new measuring technique which involves the user's thumb in the captured photo. Since volume measurement of food is the most demanding part, this technique is an important contribution to our system. Besides ease of use and

availability everywhere, in this method we know a person's thumb can be used to calibrate the image and analyze the dimensions of the selected food item. Therefore, the user will capture the photo of the selected food from the top and one side of the dish, with his/her thumb placed in the photo. Finding the volume of the food allows us to easily calculate the number of calories from the aforementioned nutritional tables, as described in the following subsection.

4.5 Nutrient Table Adoption

The structure of the food database is the main component for building and testing the food recognition system [11]. The data of the nutritional values of foods are stored in the system database in the form of tables, such as Table 1, through specialized programs. This database helps our system to calculate the number of calories in a short time and without reference to the Internet.

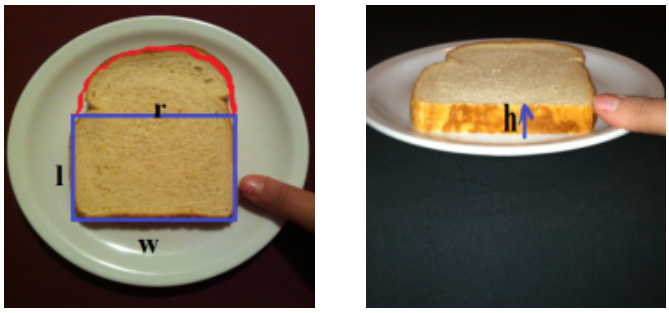
Chapter 5 Implementation

5.1 System Outline

The objective of our system is to help people who are suffering from obesity or diabetes, and people who are on a diet, to measure the daily calories they eat in each meal. The proposed system will enable mobile devices to take pictures of food for analysis and immediately respond to the user. All analysis will be supported by a set of data analysis in the system. As explained before, the proposed system consists of following steps: image acquisition, pre-processing, food segmentation, feature extraction, classification and calorie extraction. In the following sections, some of these steps will be described in more detail from an implementation point of view.

5.2 User Interface

The user captures three pictures of the food with his/her thumb in a suitable position on the dish. So, the picture will not only contain the food item, but also the user's thumb, which is used for size calibration. The two first images are taken before food consumption from the top and side view (for some food we need one picture from the side of the dish, to analyze the height of the food items in the dish). These images will enable us to extract food portions and approximate their corresponding volumes, and hence its translation to calories and nutritional facts. Figure 5-1, shows the top and side images of a food sample.



a) Top view image b) Side view image

Figure 5-1 Images from the top and side view of the food sample.

The third picture must be taken at the end of food intake, to subtract from the calculations any food not consumed by the patient. Once the food is recognized and the application suggests the type of food, the user is responsible for accepting or correcting the type of food from the application on the mobile device.

5.3 Parameterization and Measurement Process

Included as part of our proposed system, we have a practical approach for the measurement pattern, to introduce the translation of image size to real life size; in this case, the thumb of the user is the measurement pattern proposed. Figure 5-2 shows the screen for thumb calibration in the application. The dietician or the patient will also be able to introduce the dimensions of the thumb in centimeters or inches, to perform the corresponding size translation.

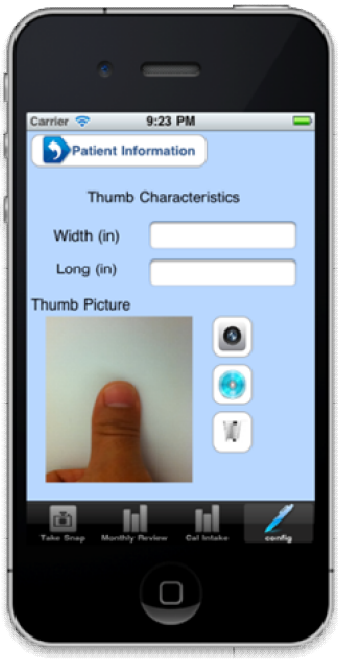


Figure 5-2: Thumb calibration image.

5.4 Segmentation

As explained in chapter 4, the goal of the segmentation step is extracting four features including: color, texture, shape and size. In this step, the thumb and its size in pixels also has to be calculated. Figure 5-3 shows a detailed flowchart of the segmentation phase. As the figure shows, many steps are common for extracting various image features. As explained before and shown in Figure 5-3, the segmentation phase starts right after performing the pre-processing step. Using a Gaussian edge detection filter, followed by a skin detection scheme, the thumb area and its size in pixels will be extracted. This size will be used in converting the pixel size of food portions to actual size. The texture features will be extracted after performing six Gabor filters on the output image of the pre-processing phase.

Extracting color, size and shape features need more processing as shown in Figure 5-3. After the pre-processing step, a color mean-shift detection phase followed by color and contour clustering steps lead to extracting all of the food portions within the original food image. For each food

portion, the color feature will be extracted, using the color histogram. The size feature will be extracted by counting the pixels in the ROI area of each food portion. The extracted size feature leads to extracting the shape feature, considering the relationship between the length and width of each ROI food portion.

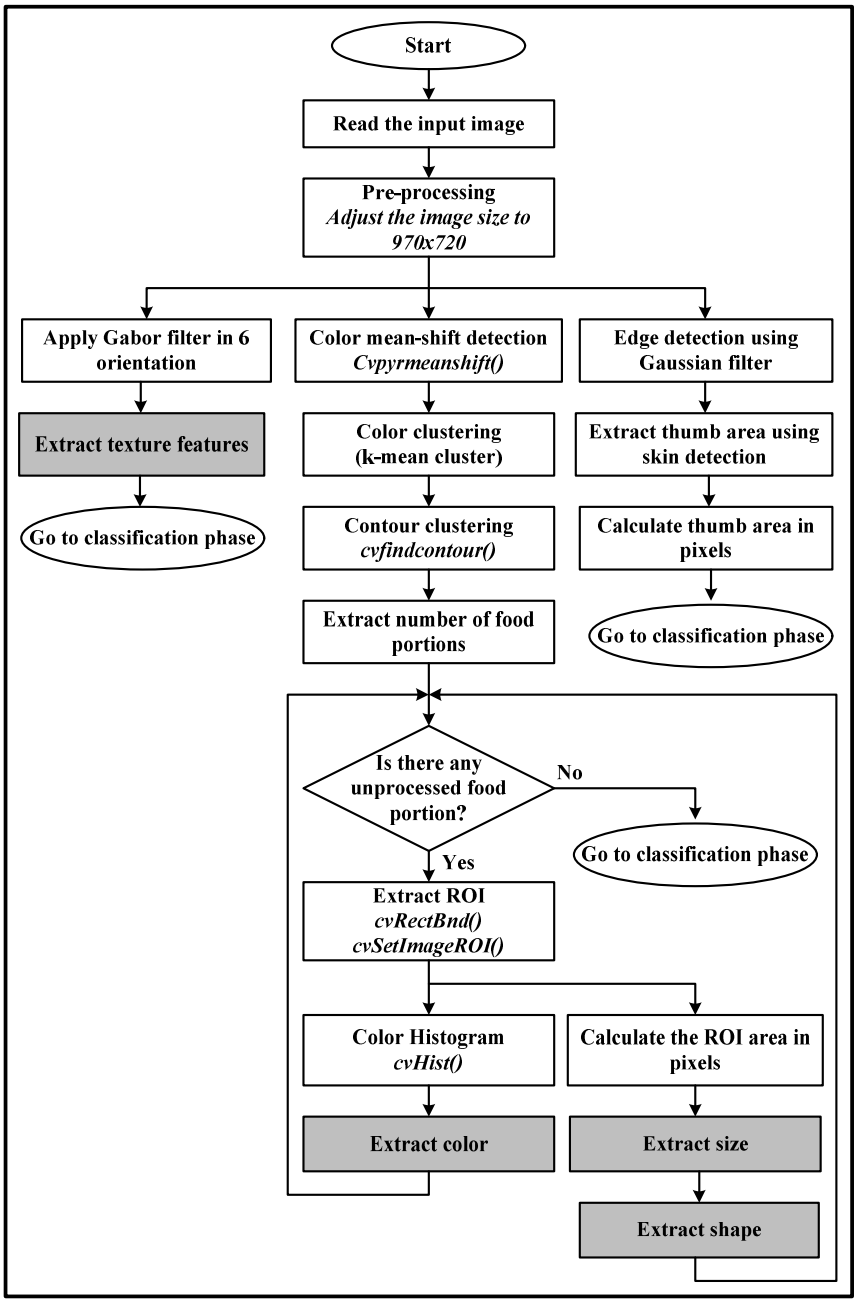


Figure 5-3 Detailed block diagram of segmentation phase.

5.5 Classification

In the classification phase, for each food portion, the extracted features will be fed into the classifier and the classifier returns the food name as its output. As explained previously, we have used the SVM technique as our classifier in this thesis. The SVM is composed of training and evaluation steps which are explained in the following subsections.

5.5.1 SVM Training Phase

Before engaging the SVM in our food recognition system, a one-time training phase has to be performed which leads to generating the SVM model.

The block diagram of the implemented training phase is shown in Figure 5-4. As the figure shows, a set of various features for each food portion is the input of this phase. Please note that the food name will also be delivered as an input in order to provide the golden output of this system. The output of this phase is the SVM model. The most important steps of Figure 5-4 are explained in the following subsections.

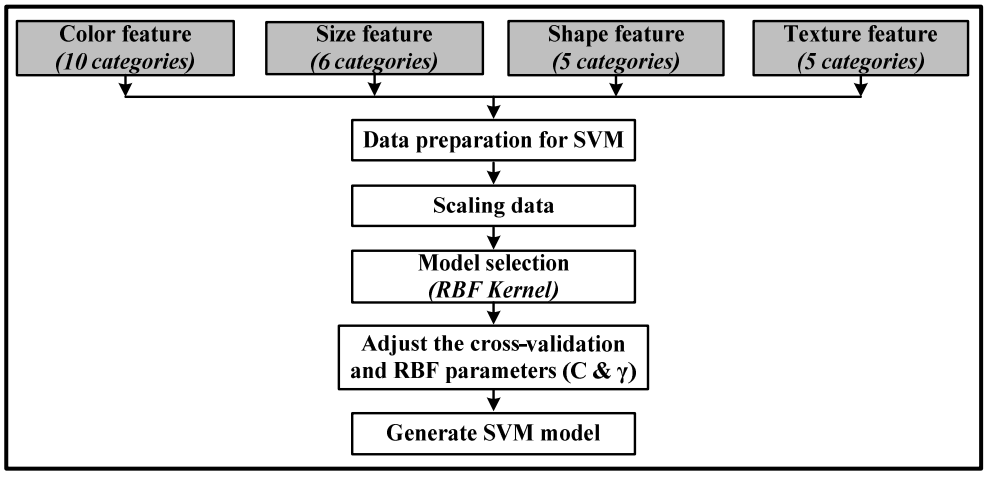


Figure 5-4 Training phase of SVM.

5.5.1.1 Data Preparation for SVM

At this step, various category combinations for each food portion are prepared. In order to have an accurate classifier we need to collect as much data as possible. This dataset should contain both positive and negative data. For example, with a food like corn, we have to prepare the combinations that represent corn (e.g. an entity with a rectangular shape, texture 1 and with a color of yellow). In addition, we have to provide the combinations which don't represent corn (e.g. an entity with a square shape, texture 2 and a color of red).

If the dataset contains only one type of data (i.e., either positive or negative combinations) the estimation process won't be accurate enough.

5.5.1.2 Scaling the data

The SVM algorithm operates on numeric attributes. So we first need to convert the data into numerical format. The original numeric values may be too large or too small in range, thus we have to rescale them to a proper range. To do so, each attribute is scaled linearly to the range of [-1; +1].

5.5.1.3 Model Selection

After scaling the dataset, we have to choose a kernel function for creating the model. Four basic kernels are defined for this step, namely: linear, polynomial, radial basis function (RBF) and sigmoid. In this project we have engaged the RBF kernel model.

5.5.1.4 Adjusting cross-validation and RBF parameters

For the RBF kernel model, the C and γ parameters have to be set, which are adjusted based on the feature values. We have used grid.py function from the *libsvm* library, which produced 2^3 and 2^{-5}

values for the C and γ parameters, respectively. The outcome of the training step is the SVM model which will be used in the evaluation phase, in order to extract the food names based on the provided features.

5.5.2 SVM Evaluation Phase

Using the SVM model, for each food portion the extracted features will be fed into the SVM system and the food name will be extracted. The whole process of this step is shown in Figure 5-5.

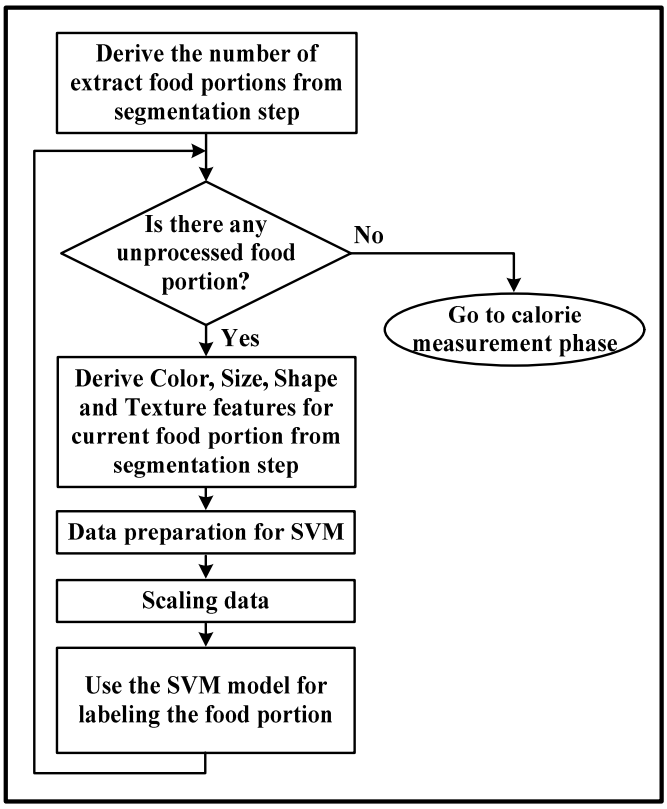


Figure 5-5 Classification phase using SVM scheme.

As shown in this figure, at first, the number of food portions extracted at the counter clustering of the segmentation phase will be derived. Based on the number of detected food portions, the SVM classification phase will be executed. For each food portion, the color, size, shape, and texture features will be sent to the SVM model. The SVM will compare these features with the feature of its

entries produced in the training step, and provide the name of the food portion. This procedure continues until all of the food portions have been labeled.

5.6 Proposed Measurement Method[3]

Having the food name extracted at the classification step, and the food portion volume extracted at the segmentation step, its caloric value can be calculated. The calorie extraction phase is explained in the following two subsections.

5.6.1 Food Portion Volume Measurement

In order to measure the size of the food inside the dish, two pictures must be taken: one from the top and one from the side, with the user's thumb placed beside the dish when taking the picture from the top. The picture from the side can be used to see how deep the food goes, and is needed for measuring the food portions' volumes. The system, which already has the dimensions of the user's thumb, can then use this information to measure the actual area of each food portion from the top picture, and can multiply this area by the depth (from the side picture) to get an estimate of the volume measurement.

To calculate the surface area for a food portion, we propose to superimpose a grid of squares onto the image segment so that each square contains an equal number of pixels and, consequently, an equal area. Figure 5-6 illustrates an example with an actual food portion. The reasons for using a grid are twofold. First, compared to other methods, the grid will more easily match with irregular shapes, which is important for food images because most food portions will be irregular. Naturally, there will be some estimation errors, but these error can be reduced by making the grid finer. Second, depending on the processing capabilities of the user's mobile device and the expected

[3] I received assistance from Rana Almaghrebi in this part

system response time from the user's perspective, we can adjust the granularity of the grid to balance between the two factors. If the grid is made finer, measurements become more accurate but will take a longer time; if the grid is made coarser, measurements become less accurate but the response time will be faster.

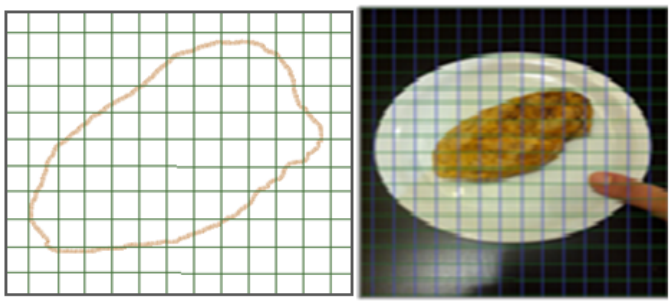


Figure 5-6 Methodology for food portion area measurement.

The total area of the food portion (TA) is calculated as the sum of the sub areas (Ti) for each square (i) in the grid, as shown in equation (39).

$$\mathrm{TA} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{T}i \tag{39}$$

Where *n* is the total number of squares in the food portion's area. After that, and by using the photo from the side view, the system will extract the depth of the food, *d*, to calculate the food portion's volume, *V*, using the following equation:

$$V = \mathrm{TA} \times \mathrm{d} \tag{40}$$

For better accuracy, if some food portions happen to be regular shapes like square, circle, triangle, etc., we can use geometric formulas to calculate their area, instead of using a grid. This however requires an additional module that can recognize regular shapes. Figure 5-7 illustrates some example calculations for regular shapes in a set of different food images.

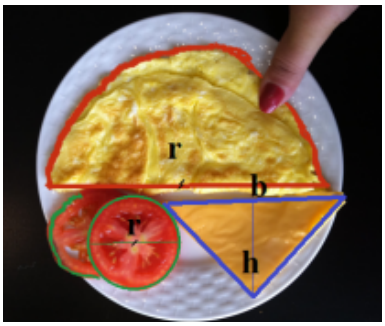


Figure 5-7 Calculating area and volume of regular shapes in food images [51].

5.6.2 Calorie and Nutrition Measurement

The volume measurement method described above is really just an interim step in order to measure the mass of the food portion. Mass is what we really need since all nutritional tables are based on food mass. Once we have the mass, we can use these tables to calculate the number of calories and other nutritional values, as described next.

It is known that the nutritional facts database is an important component for a useful and successful food recognition system. The data of nutritional values of foods are stored in these tables and are available from national and international health organizations. These tables, similar to the one shown in Table 1, help us calculate the amount of calories quickly and without reference to the Internet or an expert.

At this point, we have the measurement for the volume of each food portion, and we can use the following general mathematical equation to calculate their mass:

$$M = \rho\, V \tag{41}$$

Where M is the mass of the food portion and ρ is its density. Food density can also be obtained from readily-available tables. For example, aqua-calc provides a volume to mass conversion for 3,199 food items and ingredients [52].

After obtaining mass, the system also needs to know the type of the food, which is done by our SMV-based food recognition module. Now, the system has both the mass and the type of the food, and can use nutritional tables to lookup and measure the number of calories and other nutritional values for each food portion, as shown in equation (42).

$$calories\ in\ the\ photo\ = \frac{calories\ from\ table \times mass\ in\ the\ photo}{mass\ from\ table} \quad (42)$$

5.6.3 Partially Eaten Food

It is possible that a user does not finish all of the food captured in the first picture that was taken before eating the food. If so, we propose a simple technique to increase measurement accuracy in such cases. If a user does not finish a meal, she/he should take another top picture of what is left of the meal. All of the above process can then be repeated on this new picture to calculate the calories and nutrients in the remaining food. The actual value of food intake is then adjusted by deducting the values of the remaining food.

To test our proposed area measurement technique, we tested it on a variety of simple food (not mixed or liquid like soup, curry, etc.). We measured the area of each food portion twice: once by hand from the image, and once by using our proposed method.

Chapter 6 Evaluation

As discussed before in this thesis, we have been working with different color spaces to perform different types of analysis. One of the most important areas to highlight is the usage of the color space YCbCr, due to the capabilities attached to this color space in the skin color analysis of the pixels present inside the image. This color space made possible the thumb recognition and localization in the scene; no matter what section or location the user chose to accommodate his/her finger, we were able to extract it, and perform the corresponding measurements and translations. For the remaining part of the image which includes the food area, we used mean-shift segmentation. Because in some areas we had missing parts, we also applied the k-mean color clustering method. Overall, by combining these two methods we got more accurate results during the color segmentation stage. After that, we applied counter detection to extract the food portions and specify each part of the food by its color.

Through combining various color segmentation methods, we were able to categorize the different colors present inside each portion. For the contour extraction we were able to define, with a good level of certainty, the contours of the objects present inside the image provided by the user. Figure 6-1, shows a set of images with its contours defined by our application. The application has been tested with different types of pictures and foods present inside the images.

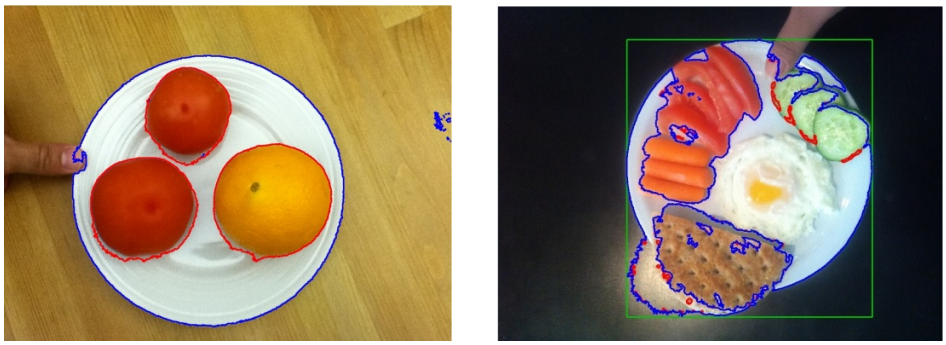


Figure 6-1 Contour images

For some images, this segmentation didn't get us good results, some examples are shown in Figure 6-2 and this encouraged us to define more features in images as we need them for the recognition stage.

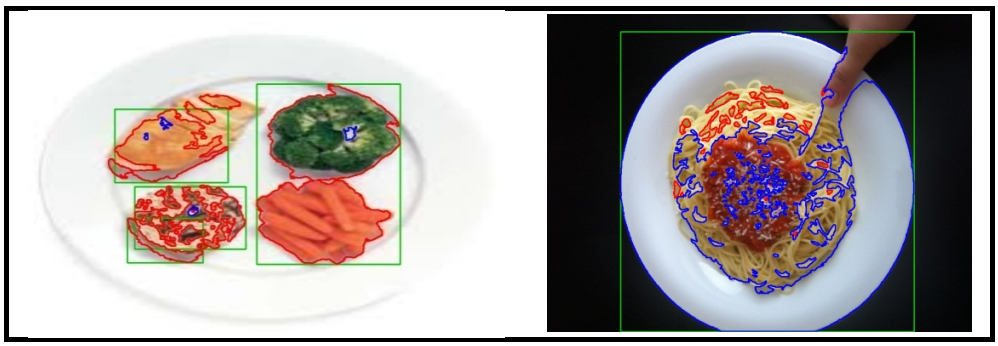


Figure 6-2 Contour images.

In order to obtain more accurate results in the segmentation stage, we also added texture segmentation to the method. For texture features, we used Gabor filters to measure local texture properties in the frequency domain. To do so, a bank of Gabor filters with different desired orientations and wavelength are applied to an image. The outcome of each of these Gabor filters is a two-dimensional array, with the same size of the input image. The sum of all elements in one such array is a number that represents the matching orientation and spatial frequency of the input image. In our method, six orientations and five preferred wavelengths are used as Gabor parameters.

Some examples of various food types and their segmented portions are shown in Figure 6-3.

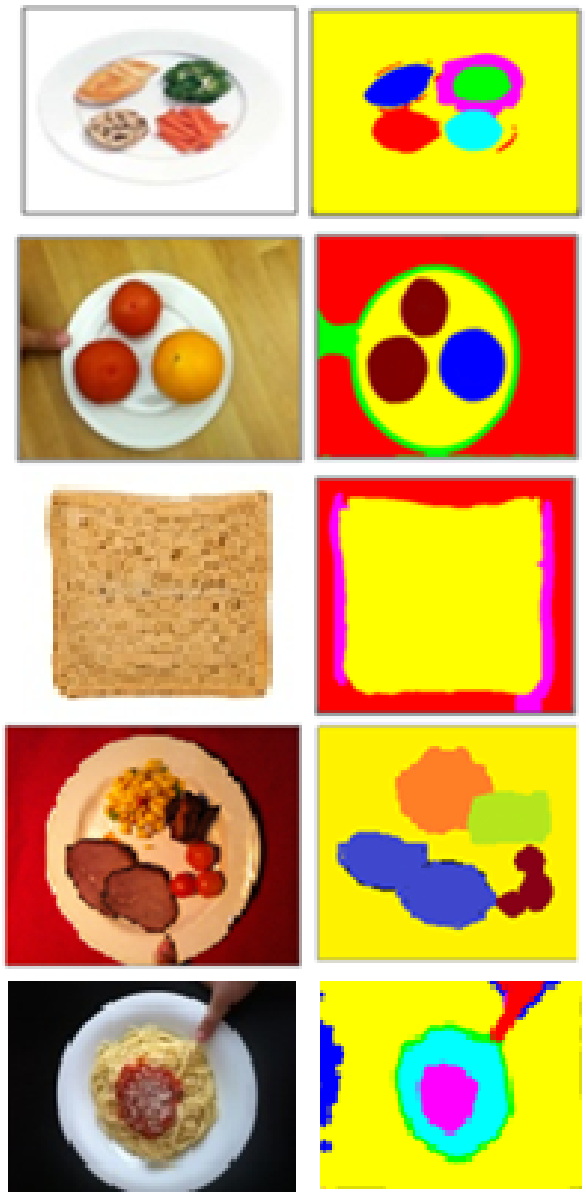


Figure 6-3 Before (left) and after (right) color and texture analysis and contour detection.

Figure 6-4 shows image analysis in some steps of the project.

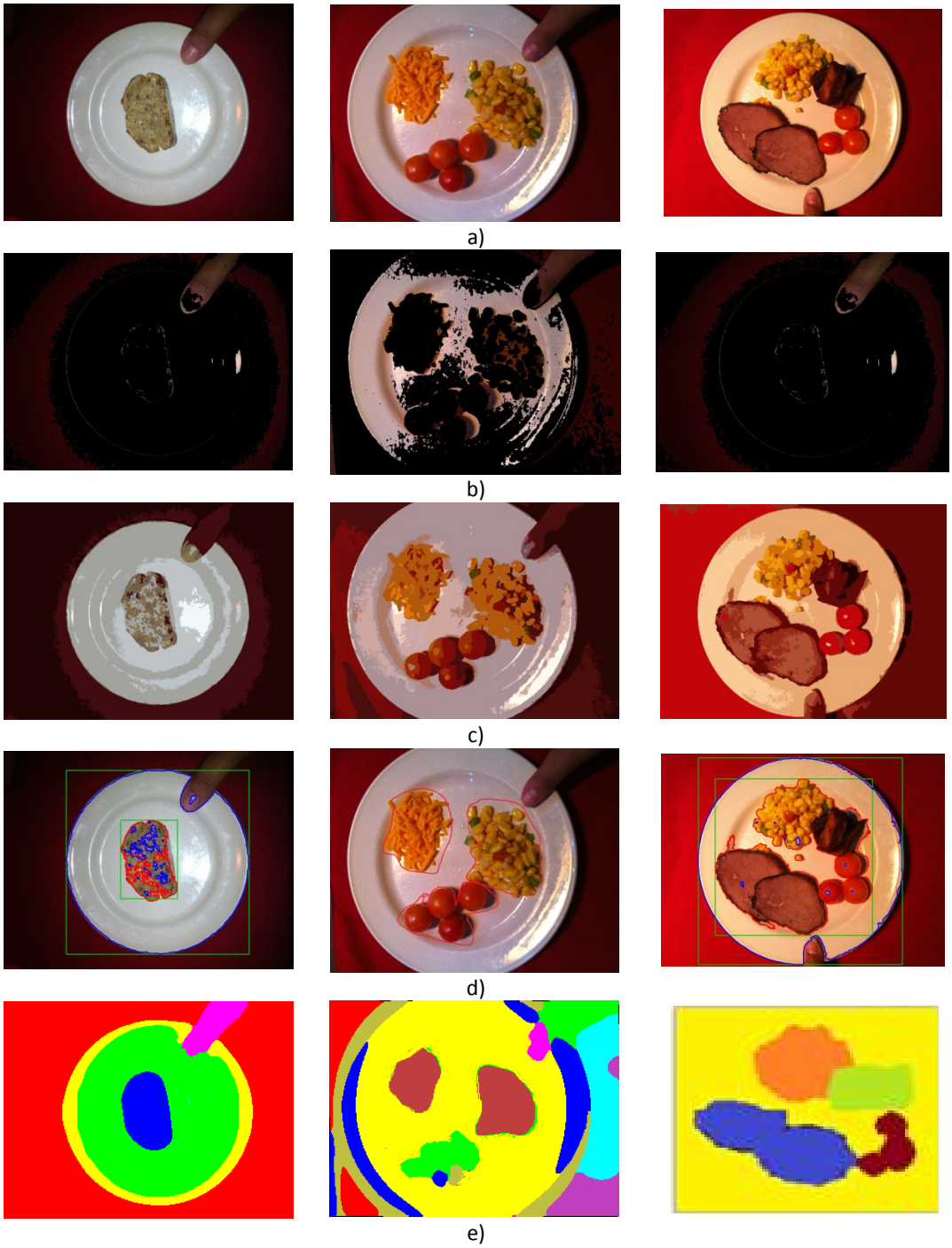


Figure 6-4 a) Original image, b) skin detection (thumb), c) color detection, d) contour detection, e) texture detection.

Once the food items are segmented and their features are extracted, the next step is to identify the food items using statistical pattern recognition techniques. Afterwards, the food item has to be classified, where we used the SVM mechanism [53], [54].

In our proposed method, the feature vectors of SVM contain five texture features, ten color features, three shape features, and six size features. The feature vectors of each food item extracted during the segmentation phase will be used as the training vectors for the SVM. Figure 6-5, shows that the SVM module verifies with the user the type of foods it has determined.

Figure 6-5 The SVM module verifies with the user the type of foods it has determined.

We have implemented our system as a software prototype, where we successfully segmented the food images and identified food portions using the contour inside of the dish [55]. We then extracted each portion one by one and analyzed them using the methods described in this thesis. For the SVM stage, we used a set of more than 200 images, approximately 100 for the training set and then another 100 images as a testing set. In the experiment, the color, texture, size and shape properties of the food images were extracted after pre-processing, as shown in Figure 6-4. We then checked the recognition result separately with the features which were color, texture, size and shape respectively. We had a low accuracy result in each feature, so we tried to test it on the joint combination of all features. The four recognition results were contrasted, and we observed that the SVM method based on color, texture, shape and size properties worked well with an accuracy of approximately 92.21 percent. The simulation results are shown in Table 2.

Table 2 Results of food and fruit recognition system.

No.	*Food items*	*Recognition Rate (%)*				
		Using Color Features	*Using Texture Features*	*Using Size Features*	*Using Shape Features*	*Using All Features*
1	Red Apple	60.33	85.25	31.22	22.55	97.64
2	Orange	65.38	79.24	41.04	71.33	95.59
3	Corn	52.00	81.93	71.33	34.61	94.85
4	Tomato	71.29	69.81	48.09	45.01	89.56
5	Carrot	74.61	79.67	69.30	65.19	99.79
6	Bread	56.11	81.56	35.55	35.20	98.39
7	Pasta	71.22	81.57	52.09	48.30	94.75
8	Sauce	72.45	78.45	40.56	55.00	88.78
9	Chicken	69.81	71.45	28.02	34.27	86.55
10	Egg	45.12	75.71	31.00	48.37	77.53
11	Cheese	61.67	83.62	42.67	33.65	97.47
12	Meat	75.38	71.67	55.00	44.61	95.73
13	Onion	45.81	79.98	31.78	22.59	89.99
14	Beans	76.80	79.55	76.71	65.11	98.68
15	Fish	58.55	64.81	18.96	62.73	77.7
16	Banana	70.02	55.00	57.67	68.05	97.65
17	Green Apple	59.33	85.25	31.22	24.76	97.99
18	Cucumber	70.02	55.00	57.67	68.05	97.65
19	Lettuce	48.11	75.71	21.00	28.37	77.55
20	Grapes	66.38	75.24	41.22	81.33	95.7
21	Potato	72.26	69.51	48.09	45.01	88.56
22	Tangerine	65.38	79.24	41.04	71.33	97.59
23	Chocolate Cake	72.17	61.56	35.55	35.20	88.19
24	Caramel Cake	54.11	61.56	35.55	35.20	85.29
25	Rice	79.39	55.62	21.40	29.00	94.85
26	Green Pepper	59.33	85.25	31.22	24.76	97.99
27	Strawberry	75.49	68.04	43.69	48.63	83.47
28	Cooked Vegetable	83.22	45.56	22.91	17.34	92.62
29	Cabbage	48.11	75.71	21.00	28.37	77.55
30	Blueberry	75.49	68.04	43.69	48.63	83.47
Total Average		**63.76**	**76.28**	**44.88**	**45.90**	**92.21**

Hence, the color, texture, size and shape information are complementary, when used together they give a good result of classification, as shown in Table 2.

As described in chapter 5, sometimes the user may not finish all of the food captured in the first picture that was taken before eating the food. If so, we propose a simple technique to increase measurement accuracy in such cases. If a user does not finish a meal, s/he should take another top picture of what is left of the meal.

To test our proposed area measurement technique, we tested it on a variety of simple foods which are listed in Table 2. We measured the area of each food portion twice: once by hand from the image, and once by using our proposed method. Our experimental results, some of which are presented in Table 3, show that our area measurement method achieves a reasonable error of about 6.39% on average.

Table 3 Area measurement experiment results.

No.	***Food items***	***Error percentage (%)***	***No.***	***Food items***	***Error percentage (%)***
1	Red Apple	2.37	16	Banana	2.2
2	Orange	4.45	17	Green Apple	2.33
3	Corn	5.15	18	Cucumber	2.01
4	Tomato	10.44	19	Lettuce	2.36
5	Carrot	0.23	20	Grapes	4.2
6	Bread	1.71	21	Potato	10.44
7	Pasta	5.22	22	Tangerine	4.32
8	Sauce	7.07	23	Chocolate Cake	10.05
9	Chicken	11.22	24	Caramel Cake	11.71
10	Egg	10.45	25	Rice	12.12
11	Cheese	2.5	26	Green Pepper	2.05
12	Meat	4.1	27	Strawberry	12.64
13	Onion	10	28	Cooked Vegetable	10.48
14	Beans	1.2	29	Cabbage	11
15	Fish	11.3	30	Blueberry	10.62
Total Error Percentage (%)			**6.39**		

Chapter 7 Conclusions

7.1 Conclusion

In this thesis, in order to increase the accuracy of mobile calorie measurement, we designed a nutrient intake measuring application that can be used on mobile devices, smart phones and Tablets. In this application, first of all the system takes an image from food. For analyzing food images, we need to use image segmentation techniques, and in this step (because we need to label each extracted part), recognition analysis was applied to the images. To increase the accuracy of recognition part, the images and the data stored in the database are used as templates to recognize the shape, color, size and texture of the images.

Now it is time to classify the extracted part of the segmenting step and for this iteration we used a support vector machine as a classifier. In the support vector machine, we have two different steps. First we have the training level, and after that the testing stage; for these steps we need two categories of images, training and testing (it is better to choose different images for each step).

After the system monitored the classifier's result, the system will ask the user to confirm whether the prediction made by the application is correct, otherwise the user can correct any mistake made in automatic prediction.

We have verified that our proposed measurement system is useful, convenient and very user-friendly, which could encourage users to use the system at each and every meal time.

In this application we have two important stages, segmentation and recognition. In this thesis, by choosing the color-texture segmentation we achieve affordable results for the recognition step and by choosing more images in each stage of training and testing for the SVM classifier, we can reach 92.2 % accuracy.

7.2 Future Work

Although the current work detects the food portions with a reasonable degree of accuracy, it can be improved by enriching various steps of the system. Another potential extension of current work is enabling the system to detect food portions within some mixed food. Such extensions need more complicated methods in all steps to increase the accuracy and reliability of the system. By considering these problems we nominated the Mobile cloud-computing (MCC) as another potential future work. In the following sections we first explain the steps that can be enriched and then we present our justification for using MCC.

7.2.1 Engaging novel segmentation schemes

For segmentation part, some novel methods including Salient Region Detection, Speeded-Up Robust Features (SURF) and also Fast Rejection, can be engaged in our future food recognition system. Similarly, for classifying phase, the RBF kernel function of SVM, can be improved by using a more accurate but more complex polynomial kernel function. In the following we will describe the mentioned novel methods for segmentation part in more detail.

7.2.1.1 Salient Region Detection

This segmentation method is using for identifying regions of interest. In our application, we are interested in image regions containing food portions. The region of interest detection is useful to assigning correct label to each pixel by rejecting non-food objects.

The first step of this method is to remove the background pixels from the food portion. Since the taken images are in color format, we can generate a foreground-background image by labeling the most frequently color in the Lab color space as the background pixel color. Also we can use the canny operator to extract the edges.

In next step, to determine the components containing potential food items, we will use the canny edge filter on each component and obtain the normalized edge histogram. We may use edge histogram for identifying food portion area. Afterwards, we will apply the Euclidean distance between the normalized edge histogram of each salient region and a regular distribution. Finally, using a threshold value, the salient region will be determined.

7.2.1.2 Speeded-Up Robust Features (SURF)

SURF is a fast and robust algorithm for local similarity invariant image comparison. SURF selects interest points of an image from the salient features of its linear scale-space, and then builds local features based on the image gradient distribution.

The SURF algorithm is composed of three following steps:

- Interest point detection
- Interest point description
- Feature matching

In the detection step, the local maxima of the Hessian determinant operator applied to the scale-space are computed to select interest point candidates.

The second step is to build a descriptor that is invariant to view-point changes of the local neighborhood of the point of interest.

The third step matches the descriptors of both images. Exhaustive comparisons are performed here by computing Euclidean distance between all potential matching pairs.

7.2.1.3 Fast rejection

Having a large number of segments does not necessarily make our method more reliable. Many segments are useless due to mis-segmentation or overlapped segmented areas. For solving this problem we will use a fast rejection step. We first filter out small segments (up to 500 pixels in area), as these segments do not contain significant feature points to represent the object classes. We then assign label to segments in each salient region which belong to background as detected.

7.2.1.4 Taste and Smell

We will try to apply the opportunity of adding taste and smell to the system. The user can add these features to the processing step, in order to have a more accurate recognition system.

7.2.2 Cloud Computing

The current proposed food recognition system, similar to other existing methods in this area, is composed of various computational intensive steps including image segmentation, feature extraction and classification. For example for color segmentation a 3×3 mask has to be convolved with the whole image. Similarly a 2×2 mask convolves with the image in case of texture segmentation. Generally, for an $M \times N$ image, and a $n \times n$ mask, $n^2 \times (M \times N)$ multiplications and $(n^2 - 1) \times (M \times N)$ addition operations are needed.

As explained before, all of these complex operations have to be performed on the mobile devices which not only have limited processing capabilities but also restricted power resources. We have observed that mobile cloud-computing is a proper solution for such a system.

Mobile cloud computing refers to a structure where both the data storage and the data processing happen outside of the mobile device. Mobile cloud applications move the computing power and data storage away from mobile phones and into the cloud.

In our desired MCC system the food image will be captured by a mobile device and the pre-processing step will be performed on it. Then the image will be sent to the server to performing the computational intensive steps such as segmentation, feature extraction, classification and calorie measurement. Finally, the estimated calorie information will sent back to the mobile device.

If we have the opportunity to use cloud computing in our application, in addition to using the server for the processing stage, it should be able to update its food database more easily since the cloud has access to all other client data. The other benefits of a mobile cloud-computing food recognition system are as follows:

7.2.2.1 Extending battery lifetime

It is clear that using the cloud computing system reduces battery consumption in mobile devices.

In order to show the effect of using MCC in our food recognition system, we have extracted the complexity of a mobile device in two cases. In the first case, our current conventional food recognition system is considered, while in the second one the MCC system is engaged. In the first case, all recognition steps are performed on mobile devices, while in the second case just the image acquisition and pre-processing steps are performed on mobile devices and other steps will be

performed in the cloud. The number of clock cycles in each case, for a food image of 970×720 size, extracted using Intel V-tune software [56] is shown in Table 4, below.

Table 4 **T**he number of clock cycles needed for a mobile device in two different cases.

Case	Tasks	Number of clock cycles needed
1 - Conventional method	Whole food recognition steps	1,346,000,000
2 - Mobile cloud computing	- Image acquisition - Pre-processing	460,000,000

As shown in Table 4, the computational complexity (number of clock cycles) in the second case is one third of the complexity of the first case. Since power consumption has a linear relation with system workload [57], the power consumption of the second case will be one third that of the first case.

7.2.2.2 Improving data storage capacity

Storage capacity is also a limitation for mobile devices. Mobile cloud computing is developed to enable mobile users to store and access large amounts of data in the cloud through wireless networks.

In our case, the whole system image database can be stored in the server side and just the initial and final images will be stored on mobile devices. In addition, users can save considerable amounts of energy since no effort is made by mobile devices to store the database and internal image data.

Also for Recognition step, If we have enough capacity to save more images we will get better results from classification level. As we know by using more images in training phase we can train the system more accurately and this step helps us to have better output in testing phase, so applying cloud computing helps us to have better results.

7.2.2.3 Improving reliability

Storing data or running applications in the cloud is an effective way to improve reliability since both data and application are stored and backed up on a number of computers. This reduces the chance of data and applications being lost from mobile devices.

References

[1] K. Flegal, M. Carrol, C. Johnson C. Ogden, "Prevalence and trends in overweight among US children and adolescents, 1999-2000," *JAMA*, vol. 288, pp. 1728–1732, October 2002.

[2] J. Saadinem, K. Flegal, G. Beckles A. Fagot-Campagna, "Diabetes, impaired fasting glucose, and elevated hba1c in US adolescents," *the third National Health and Nutrition Examination Survey," Diabetes Care"*, vol. 24, pp. 834–837, 2001.

[3] M. Livingstone, J. Wallace, and P. Robson, "Issues in dietary intake assessment of children and adolescents," *Br.J.Nutr*, vol. 92, pp. 213–222, 2004.

[4] Y. C. Probst and L. C. Tapsell, "Overview of computerized computerized dietary assessment programs for research and practice in nutrition education," *J. Nutr. Educ. Behav*, pp. 20–26, 2005.

[5] R. Johnson, R. Soultanakis, and D. Matthews, "Literacy and body fatness are associated with underreporting of energy intake in US low-income women using the multiple-pass 24-hour recall: a doubly labeled water study," *J.Am.Diet.Assoc*, vol. 98, pp. 1136–1140, 1998.

[6] D. H. Wang, D. H. Kogashiwa, and S. Kira, "Development of a new instrument for evaluating individuals' dietary intakes," *J. Am. Diet. Assoc. 106*, pp. 1588–1593, 2006.

[7] L. Harnack, L. Steffen, D. Arnett, S. Gao, and R. Luepker, "Accuracy of estimation of large food portions," *J.Am.Diet.Assoc.*, vol. 104, pp. 804–806, 2004.

[8] R. C. Klesges, L. H. Eck, and J. W. Ray, "Who underreports dietary intake in a dietary recall? Evidence from the Second National Health and Nutrition Examination Survey," *Consult.Clin*

Psychol, vol. 63, pp. 438–444, 1995.

[9] F. E. Thompson and A. F. Subar, *Dietary Assessment Methodology*, 2nd ed.: Nutrition in the Prevention and Treatment of Disease., 2001.

[10] "USDA food and nutrient database for dietary studies, 1.0.," Beltsville, MD: Agricultural Research Service, Food Surveys Research Group, 2004.

[11] M.i Sun et al., "Determination of food portion size by image processing," in *30th Annual International IEEE EMBS Conference*, 2008, pp. 871–874.

[12] C. K. Martin, S. Kaya, and B. K. Gunturk, "Quantification of Food Intake Using Food Image Analysis," *Conf Proc IEEE Eng Med Biol Soc.*, pp. 6869–6872, 2009.

[13] P. Y. Chi, J. H. Chen, H. H. Chu, and J. L. Lo, "Enabling Calorie-Aware Cooking in a Smart Kitchen," *3rd international conference on Persuasive Technology*, 2008.

[14] M. S. Westerterp-Plantenga, "Eating behavior in humans, characterized by cumulative food intake curves—a review," *Neuroscience and Biobehavioral Reviews*, vol. 24, pp. 239–248, March 2000.

[15] K. Chang et al., "The diet-aware dining table: Observing dietary behaviors over a tabletop surface," *Proceedings of Pervasive Computing — 4th International Conference*, pp. 366–382, May 2006.

[16] J. Chae et al., "Volume Estimation Us-ing Food Specific Shape Templates in Mobile Image-Based Dietary Assessment," in *Proceedings of the IS&T/SPIE Conference on Computational Imaging IX*, Burlingame, California, 2011, p. 275—281.

[17] F. Zhu, M. Bosch, N. Khanna, C. J. Boushey, and E. J. Delp, "Multilevel Segmentation for Food Classification in Dietary Assessment," in *Proceedings of 7th International Symposium on Image*

and Signal Processing and Analysis, Dubrovnik, Croatia, 2011, p. 337—342.

[18] S. Arivazhagan, R. N. Shebiah, S. S. Nidhyanandhan, and L. Ganesan, "Fruit Recognition using Color and Texture Features," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 1, p. 90—94, October 2010.

[19] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed.: Pearson Prentice Hall, 2008.

[20] T. F. Chan and L. A. Vese, "Active Contours Without Edges," *IEEE Transactions on Image Processing*, vol. 10, no. 2, February 2001.

[21] J. Zhang, T. Tan, and L. Ma, "Invariant texture segmentation via circular Gabor filters," *Proceedings of the 16th IAPR International Conference on Pattern Recognition (ICPR)*, vol. Vol II, p. 901—904, 2002.

[22] B. Kartikeyan and A. Sarkar, "An identification approach for 2-D autoregressive models in describing textures," *Graphical Models and Image Processing*, vol. 53, p. 121—131, 1993.

[23] R. M. Haralick, K. Shanmugan, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 3, p. 610—621, 1973.

[24] A. Jain and G. Healey, "A multiscale representation including opponent color features for texture recognition," *IEEE Transactions on Image Processing*, vol. 7, no. 1, p. 124A multiscale representation including opponent color features for texture recognition128, 1998.

[25] B. Julesz, "Textons, the fundamental elements in preattentive vision and perception of textures," *Bell Syst. Tech. J*, vol. 62, no. 6, pp. 1619–1645, 1983.

[26] J. R. Bergen and M. S. Landy, *Computational modeling of visual texture segregation*.: MIT Press, 1991.

[27] P. Kruizinga and N. Petkov, "Non-linear operator for oriented texture," *IEEE Trans. on Image Processing*, vol. 8 (10), pp. 1395–1407, 1999.

[28] S. E. Grigorescu, N. Petkov, and P. Kruizinga, "Comparison of texture features based on Gabor filters," *IEEE Trans. on Image Processing*, vol. 11 (10), pp. 1160–1167, 2002.

[29] L. G. Roberts, "Machine perception of 3-D solids," *Optical and Electro-Optical Information Processing. MIT Press*, pp. 159–197, 1965.

[30] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, 1986.

[31] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active Contour Models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1988.

[32] G. Villalobos, "Food Image Processing for a Semi-Automatic Nutrient Intake Monitoring System," University of Ottawa, Otawa, thesis 2012.

[33] H. Wang and S. F. Chang, "A Highly Efficient System for Automatic Face Region Detection in MPEG Video," *IEEE Transactions on Circuits and Systems for Video Thechnologies*, vol. 7, no. 4, 1997.

[34] A. Albiol, L. Torres, C. A. Bouman, and E. J. Delp, "A Simple and efficient face detection algorithm for video database applications," 2000.

[35] W. Jia et al., "A food portion size measurement system for image-based dietary assessment," *Bioengineering Conference, IEEE*, pp. 3–5, April 2009.

[36] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. San Diego: Ca: Academic Press, 1990.

[37] R. Duta, P. Hart, and D. Stork, *Pattern Classification*. New York: NY: Wiley-Interscience, 2000.

[38] V. Vapnik, "The Nature of Statistical Learning Theory," *Springer*, 1995.

[39] C. Burges, *A tutorial on support vector machines for pattern recognition*.: Kluwer Academic Publishers, Boston, 1998, vol. 2.

[40] V. Vapnik, S. Golowich, and A. Smola, "Support vector method for function approximation, regression estimation, and signal processing," in *Advances in Neural Information Processing Systems*, Cambridge, MA, 1997, pp. 281–287.

[41] D. M. Skapura, *Building Neural Networks*.: ACM press, 1996.

[42] S. Keerthi and C. J. Lin, "Asymptotic behaviors of support vector machines with Gaussian kernel," *Neural Computation*, vol. 15, pp. 1667–1689, 2003.

[43] B. Schölkopf, A. Smola, R. Williamson, and P. L. Bartlett, "New support vector algorithms," *Neural Computation*, vol. 12, no. 5, pp. 1207–1245, May 2000.

[44] B. Schölkopf, J. Platt, J. Shawe, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443-1471, July 2001.

[45] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*.: Cambridge University Press, 2000.

[46] T. Mitchell, "Machine Learning," *McGraw-Hill Computer Science Series*, 1997.

[47] C. C. Chang and C. J. Lin, "LIBSVM : a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology 2:27:1–27:27*, 2011.

[48] P. Kruizinga, N. Petkov, and S. E. Grigorescu, "Comparison of texture features," in *Proceedings of the 10th International Conference on Image Analysis and Processing*, Washington DC, USA,

September 1999, pp. 142–148.

[49] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognition*, vol. 24, pp. 1167–1186, 1991.

[50] G. Villalobos, R. Almaghrabi, B. Hariri, and S. Shirmohammadi, "A Personal Assistive System for Nutrient Intake Monitoring," in *International ACM Workshop On Ubiquitous Meta User Interfaces*, 2011, pp. 17–22.

[51] R. Almaghrabi, G. Villalobos, P. Pouladzadeh, and S. Shirmohammadi, "A Novel Method for Measuring Nutrition Intake Based on Food Image," in *Proc. IEEE International Instrumentation and Measurement Technology Conference*, Graz, Austria, 2012, pp. 366–370.

[52] W. Wu and J. Yang, "Fast Food Recognition From Videos of Eating for Calorie Estimation," in *Intl. Conf. on Multimedia and Expo*, 2009.

[53] K. Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf, "An introduction to kernel-based learning," *IEEE Transactions on Neural Networks*, vol. 12, no. 2, pp. 181–201, March 2001.

[54] V. Vapnik and C. Cortes, "Support-vector network," *Machine Learning*, vol. 20, pp. 273–297, 1995.

[55] P. Pouladzadeh, G. Villalobos, R. Almaghrabi, and S. Shirmohammadi, "A Novel SVM Based Food Recognition Method for Calorie Measurement Applications," in *Proc. International Workshop on Interactive Ambient Intelligence Multimedia Environments, in Proc. IEEE International Conference on Multimedia and Expo*, Melbourne, Australia, 2012, pp. 495–498.

[56] Intel. (2012) Intel VTune Performance Analyzer. [Online]. http://software.intel.com/en-us/articles/intel-vtune-amplifier-xe/

[57] Z. He, Y. Liang, L. Chen, I. Ahmad, and D. Wu, "Power-rate-distortion analysis for wireless video communication under energy constraints," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 5, pp. 645–658, 2005.