2010

Quantification of energy intake using food image analysis

Sertan Kaya
Louisiana State University and Agricultural and Mechanical College

QUANTIFICATION OF ENERGY INTAKE USING FOOD IMAGE ANALYSIS

A Thesis

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering

in

The Department of Electrical and Computer Engineering

by

Sertan Kaya
B.S., Gebze Institute of Technology, 2005
August 2010

To my dear parents: Nurhayat and Yasar

ACKNOWLEDGMENTS

I would like to thank Dr. Bahadir K. Gunturk for his endless support, brilliant inspiration and priceless guidance during my Master of Science education. I would like to thank Pennington Biomedical Research Center (PBRC) and Dr. Corby K. Martin for allowing me use the Baylor food images and giving me chance to get involved in the Bioengineering Group. I also thank to fellow student Robert for his remarkable assistance during my research path.

Special thanks to Dr. Suresh Rai and Dr. Jianhua Chen for precious knowledge that they lectured in neural computing and machine learning classes, as well as serving in my committee. I also appreciate to National Institute of Health (NIH) and National Science Foundation (NSF) for funding this research. My special thanks go to my roommates, Cihan Uzmanoglu and Mehmet Burak Cil as well.

TABLE OF CONTENTS

ACKNOWLEDGMENTS iii

ABSTRACT vi

1. INTRODUCTION 1

2. LITERATURE REVIEW 5
2.1. Methods to Measure Energy Intake 5
2.1.1. Doubly-Labeled Water (DLW) and Self-Report Methods 5
2.1.2. The Digital and Food Photography 6
2.1.3. Computer-Based Applications via Hand-held Devices 7

3. REFERENCE CARD DETECTION AND GEOMETRIC TRANSFORMATIONS 12
3.1. Reference Card Detection 12
3.2. Geometric Correction 14
3.2.1. Affine Transformation 18
3.2.2. Perspective Transformation 19
3.2.3. Scaling 21
3.3. Photometric Correction 21

4. IMAGE ANALYSIS 26
4.1. Mahalanobis Distance 26
4.2. Multilayer Neural Networks and the Backpropagation Algorithm 27
4.2.1. Introduction 27
4.2.2. Feedforward Operation 28
4.2.3. The Backpropagation Algorithm 30
4.3. Training Stage 33
4.3.1. Region Selection and Feature Extraction 33
4.3.2. Training of the Classifiers 38
4.3.2.1. Training of the Mahalanobis Distance 38
4.3.2.2. Training of the Neural Network 39
4.4. Testing Stage 40
4.4.1. Pixel Classification 41
4.4.1.1. Pixel Classification with Mahalanobis Distance 41
4.4.1.2. Pixel Classification with Neural Network 41
4.4.2. Segmentation 42

5. ESTIMATION AND PERFORMANCE ANALYSIS 53
5.1. Gram Estimation and Performance analysis 53

6. CONCLUSIONS AND FUTURE WORK 59

REFERENCES 60

VITA 63

ABSTRACT

Obtaining real-time and accurate estimates of energy intake while people reside in their natural environment is technically and methodologically challenging. The goal of this project is to estimate energy intake accurately in real-time and free-living conditions. In this study, we propose a computer vision based system to estimate energy intake based on food pictures taken and emailed by subjects participating in the experiment. The system introduces a reference card inclusion procedure, which is used for geometric and photometric corrections. Image classification and segmentation methods are also incorporated into the system to have fully-automated decision making.

1. INTRODUCTION

The prevalence of obesity is getting increased dramatically in the United States [1, 2]. Obesity is one of the most leading factors for numerous diseases, such as heart disease, hypertension and diabetes [3]. There is a strong correlation between obesity and positive energy balance, which is the difference ingested energy from expended energy. Energy Intake (EI) is being taken into account as one of the primary reasons for gaining weight. Energy intake can be defined as the calorie equivalent of the consumed amount. Measuring free-living peoples' EI presents methodological and technical challenges. Even though extensive research has been conducted in measuring EI, accurate and affordable methods have not been proposed yet [4].

In this study, a unique framework is proposed specifically to measure food intake in free-living conditions, and it builds upon the digital photography methodology [5, 6]. The proposed food intake evaluation system consists of reference card detection and geometric transformations, image analysis, gram amount estimation, and manual correction modules.

In Section 3, we articulate how to the detect reference card and its corners and then utilize it in geometric and photometric corrections. The reference card corners are used to geometrically correct the image to account for the view-angle and distance of the camera to the food. The color of the reference card is used to do photometric correction, which is essential to extract reliable color features to be used in classification and segmentation.

The purpose of the image analysis section (Section 4) is to explain how the classification and segmentation techniques are incorporated into the system. The classification module requires a training stage, where the food region is outlined by user; this is employed to extract the features for each food type. The training stage is illustrated in Figure 1.1. *The Mahalanobis distance* [7] and *multilayer neural networks* [36] are the techniques incorporated into the classification

module. In the testing stage, pixel classification is followed by segmentation to determine the food regions. There is an optional manual correction in case of the automatic classification and segmentation processes fail. The flowchart of the testing stage is illustrated on Figure 1.2.

Section 5 details how to obtain gram amounts from classified and segmented food images. The classified/segmented image is warped to do geometric correction. The real area of food region is determined using the reference card. The gram amount for each food type, which is entered in the training stage, has a correlation with food region area. Based on the association between the food region area and gram amount, the amount of the food in the picture is estimated. Once we have the estimated gram amount, this value is combined with the portion code and portion weight as well as the nutritional information in the USDA Food and Nutrient Database for Dietary Studies (FNDDS) to determine the final energy intake.

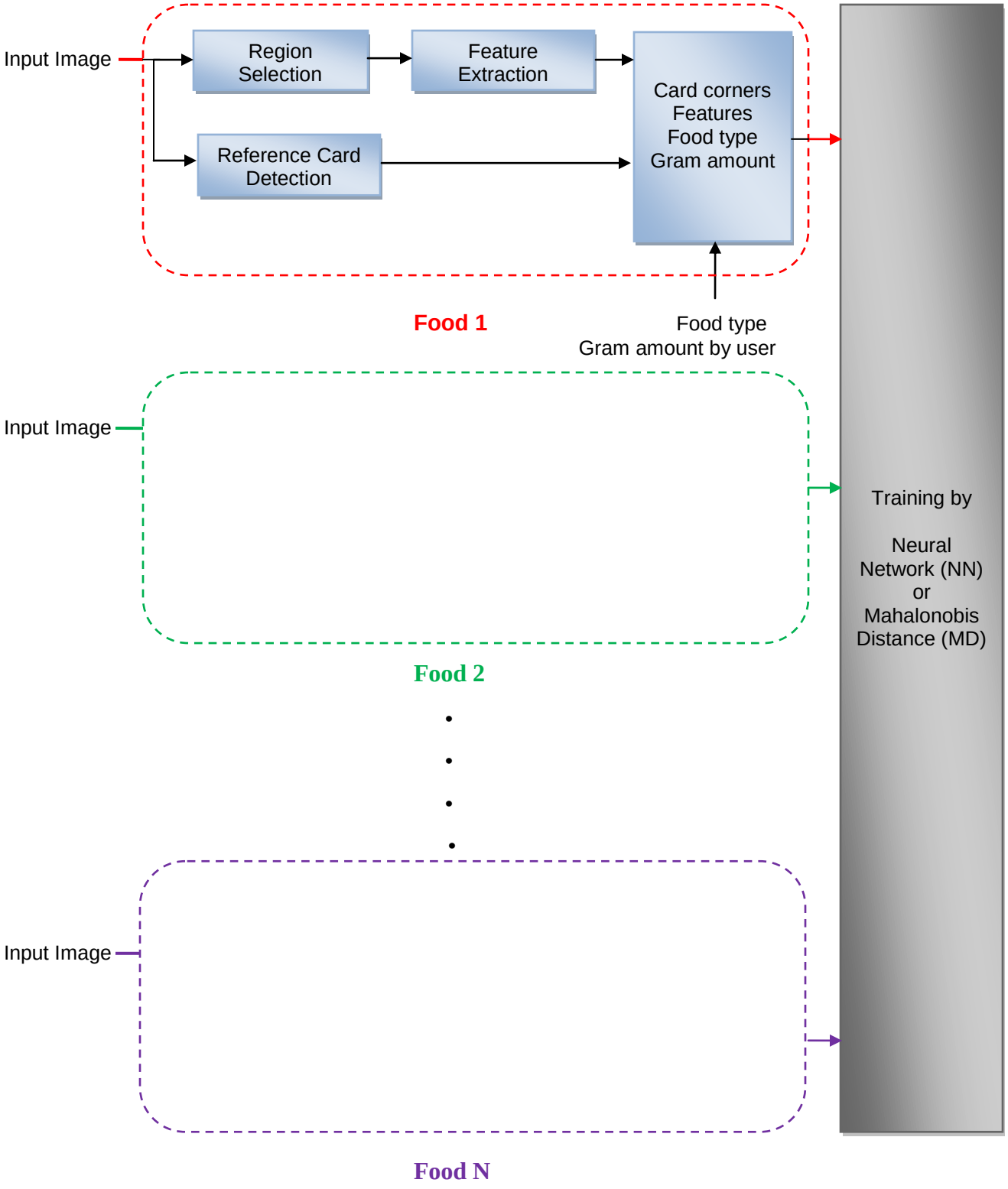


Figure 1.1: Training stage of the system.

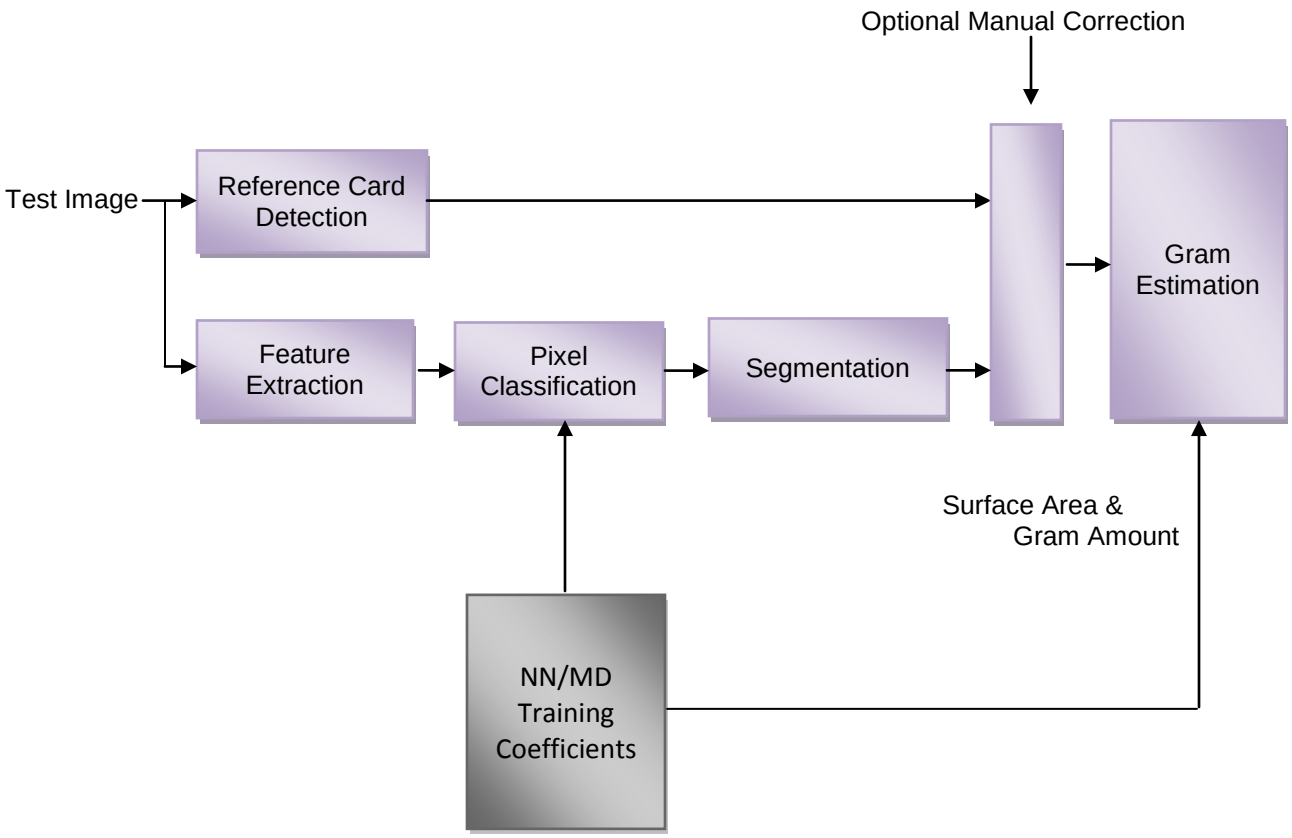


Figure 1.2: Testing stage of the system.

2. LITERATURE REVIEW

2.1. Methods to Measure Energy Intake

2.1.1. Doubly-Labeled Water (DLW) and Self-Report Methods

Obtaining and measuring energy intake accurately in real-time is a challenging task while people reside in their natural environment. Currently, the doubly-labeled water (DLW) is known as the gold standard for measuring energy intake in free-living conditions. The fundamental idea behind this technique is explained as follows [10, 11]. A water solution that contains deuterium and oxygen is given to the subject, and the urine samples are gathered for analysis. In the body, the deuterium is fluxed and turned into water, whereas oxygen is fluxed and transformed into water and carbon dioxide. The difference of the rates of transformation results, from deuterium to water and from oxygen to water-carbon dioxide, is a measure of carbon dioxide flux. With this carbon dioxide flux, energy expenditure can be calculated using standard indirect calorimetric equations. The DLW consists of some experimental analysis, applications of indirect calorimetric, calculation methods and some complicated assumptions and sources of errors. The DLW accounts for the total energy expenditure (TEE), and during energy balance, energy expenditure is equivalent to energy intake. However, it is the fact that when a person is experiencing a large energy deficiency, energy balance will not equal to energy intake; this will cause some difficulties for obtaining an accurate estimate of energy intake [12]. Even though the DLW gives us an objective measure of energy intake, it requires some intricate experimental steps and drives a high-cost. Another drawback of the DLW method is that it does not provide any information regarding ingested composition of foods. As a result, the DLW has some limitations in order to obtain accurate energy intake. Hence, it is not an applicable method which can be simply used as a tool in free-living conditions.

Most alternative methods to measure food intake rely on participant self-report, including food records, 24-hour recall, and food frequency questionnaires. These methods rely on the participants to recall or record their food consumption and estimate or measure of the amount (portion) of food eaten. Although these methods are frequently utilized to estimate food intake in research and clinical settings, they underestimate food intake by 37% or more [13-15]. In addition to this, people who are overweight or obese underreport food intake to a greater degree than lean people when using these methods [15]. The largest source of error in estimating food intake from self-report is attributable to participants' poor estimation of portion size [16]. Hence, methods that do not rely on the participant to estimate portion size are needed.

2.1.2. The Digital and Food Photography

The aim of food photography is to assist the subject to estimate portion size by utilizing depicted portion size ranges in the photograph [17]. The purpose of the research is to determine the correlation between food photography and estimation, and the errors related with the conceptualization. In this study, volunteered people from different social and professional backgrounds were instructed to complete evaluations regarding estimating portion size associated with photographs. Subjects were asked to eat one meal (breakfast, lunch, or dinner) consisting of four and six various food types. Portion sizes were weighted before and after the meal by the researchers. After five minutes passed, researchers began to show food photographs depicting portion size incrementing from 5^{th} to the 95^{th} percentile. Following these photographs, subjects were questioned to determine the portion size correlated with consumed food and nutrient content of meals. In the result of the questionnaire, apparently small portion sizes were overestimated whereas larger portion sizes were underestimated. Consequently, the food photography, which depicts various ranges of food portion sizes, was a beneficial method for the subject who was asked to estimate what was consumed. This research also revealed the errors

where it occurred from the sense of conceptualization. Additionally, body mass index, age, sex and portion size are critical factors during the food intake estimation.

The digital photography of foods method [18, 19] was developed to unobtrusively measure food intake in cafeteria settings, and it does not rely on the participant to estimate portion size. The digital photography of foods method involves using a digital video camera to capture a photograph of a participant's food selection before they eat, and plate waste after they finish eating. While in the cafeteria or dining location, photographs are carefully captured of measured standard portions of the foods served on the day of data collection. At a later date in the laboratory, these photographs are analyzed by registered dietitians (RDs) who estimate the amount (portion) of food selection and plate waste by comparing these photographs to the standard portion photographs. These portion size estimates are entered into a custom built computer application that automatically calculates the grams, kilocalories (kcal), and macro- and micro-nutrients of food selection, plate waste, and food intake based on a United States Department of Agriculture (USDA) database [20]. The digital photography of foods method has been found to be highly reliable and accurate (valid), and overestimates energy intake by less than 6 grams on average [19].

2.1.3. Computer-Based Applications via Hand-Held Devices

With rapidly developing technology, hand-held devices such as PDA (Personal Digital Assistant) have become the part of energy intake dietary analysis system in which most of them consist of computer-based applications and hand-held devices. The PDA-based energy intake program is aimed to assess the validity of the usage of PDA as a food recording tool and investigate source of error from this methodology [16]. In study, after the subjects were trained how to use PDA as the food recorder tool, they were let to use PDA for 72 hours to test the usability and to get preliminary food records. Aftermath, they were given a questionnaire,

regarding the usability of PDA, and 24-h recall. The subject used PDA to obtain food records for an observed and weighted lunch while recording time was kept for each meal by researchers. The actual and 24-h recall food intake results were compared with recorded intakes where the subjects used PDA. In this comparison, PDA-as the food recording tool and 24-h recall were found as fairly correlated, an exploration of the source errors showed that food portion estimation was primary source of the error. Finally, PDA-based food recording method is comparable to 24-h recall in the sense of energy intake.

Another PDA-based energy intake method was introduced by utilizing an innovated hand-held device consisting of a digital camera and a mobile phone card attachment. In the proposed paper by Wang et al. [21], the validity study in which the majoring food and nutrition subjects were asked to take pictures of recorded pictures while having 1-d weighed diet records, and they sent them to the dietitians by using mobile phone card attached PDA. The reliability in which the subjects were asked to take the pictures of the meal by two instruments, and they sent them to the different dietitians via mobile phone card capability. In order to assist dietitians to estimate accurately, the subjects were required to apply these essential four steps while taking digital images as follows. The ruler-stick should be placed next to the plate and the PDA was held at 45 angled while taking pictures. The content of the food types was typed into PDA and added details of the subject dietary habits. With regard to the validity, the innovated PDA method gave the comparable result with respect to the diet record method. This research concluded that the innovated PDA based method was beneficial tool for estimating energy intake. One of the most recent novel approaches in the hand-held-based energy intake system was studied as follows [22]. What the proposed system aimed is that a network connected PDA, which has built-in camera on it, is employed to take digital images during before and after intake, and with attaching some specific information such as time label, send them to the central nutrient

database consisting of automatic image analysis and visualization system. The case of where the food image can't be captured or be found, PDA let the used to search and find the food image from existing menu. The automatic image analysis and visualization is composed of image calibration and acquisition, segmentation, feature extraction, classification, volume estimation and computing consumed food steps. The goal of image calibration and acquisition step is to obtain 3D calibrated system by having reference information such as the dimension of the plate and different angled views. In the segmentation step, the food regions are automatically segmented from the image scene by first converting image to grayscale then YCbCr color space. Color and Gabor texture features are considered to perform feature extraction step. After performing segmentation and extracting the features, statistical pattern recognition techniques [23, 24] are employed to do food recognition. Support vector machine SVM [25] is the classifier which was used in the classification step. The volume estimation step is indicated as a future work in which multiple images and 3D shape reconstruction techniques will be devised to use. From before and after meal images, obtaining volume will be integrated with portion code and weight in the USDA Food and Nutrient Database for Dietary Studies (FNDDS) to compute the gram of the particular food type.

This proposed research was used as a pilot study in children's dietary assessment [26]. In the pilot study, the participants, Chinese girls and boys, were asked to use six different dietary assessment methods: 24-hr, FR (paper and pencil), PDA with hierarchical menu, PDA with search menu, PDA with camera, camera with notebook. After they applied these methods, they were given questionnaire to obtain their feedback regarding these methods. The results showed that adolescents had much more tendency for using digital images and PDA-based tools than other methods.

Mariappan et al. [27] presented the updated status of Zhu et al. [22] where it was stated that the dietary assessment system consisted of two different configurations, namely client-server and standalone configuration. In the client-server configuration, the client takes the picture of the food and sends them to the server where the image analysis is performed, food recognition and volume estimation. The server then sends back to the client with the results to obtain confirmation. Lastly, the confirmed data gets back to the server for matching nutrition and using further analysis. In the standalone configuration, all steps are performed in the mobile device. However, it was indicated that it was not implemented. In the classification results, they obtained 93.745% classification accuracy by using Food Replicas where the images were obtained under certain conditions such as the white plate and the checker-board patterned tablecloth were used to help segmentation and volume estimation. On the other hand, they also obtained 57.55% classification accuracy when real foods were used on the checker-board patterned tablecloth. Another progress they described in this paper is the frame of the user interface where the user has opportunity to label or edit the meal, and the user interface lets also the user intervene when food can't be recognized.

The Remote Food Photography Method (RFPM) [28, 4] was developed specifically to measure food intake in free-living conditions and it builds upon the digital photography methodology. When using the RFPM, participants are trained to use a camera-equipped cell phone with wireless data transfer capabilities to take pictures of their food selection and plate waste and to send these pictures to the researchers over the wireless network. To reduce the frequency of participants forgetting to take photographs of their foods, they receive and respond to automated prompts reminding them to take photographs and to send the photographs to the researchers. These prompts are consistent with ecological momentary assessment (EMA) [29] methods and consist of emails and text messages. The images are received by the researchers in

near real-time and can be analyzed quickly to estimate food intake. When analyzing the images, the RDs rely on methods similar to the digital photography of foods method, but the participants are not required to take a photograph of a standard portion of every food that they eat, as this would be unfeasible. Alternatively, an Archive of over 2,100 standard portion photographs was created. This allows the RDs to match foods that participants eat to a standard portion photograph that already exists. Initial tests supported the reliability and validity of the RFPM [28, 4]. The RFPM underestimated food intake by ~6% and, importantly, the error associated with the method was consistent over different levels of body weight and age.

Woo et al. [30] articulated the study of Zhu et al. [22] where volume estimation was promised as a future study. With the fact that the image itself and segmented image are employed, as well as the checker-board patterned card are placed next to every meal for performing scaling and posing, the camera parameters, which are essential for the further estimation, are obtained. Feature points are extracted and unprojected onto 3D space. In the volume estimation approach, utilizing camera parameters and feature points, spherical and prismatic approximation models are applied to reconstruct of various food types. Lastly, visual refinement is carried for prismatic objects.

3. REFERENCE CARD DETECTION AND GEOMETRIC TRANSFORMATIONS

3.1. Reference Card Detection

In order to estimate food portions accurately in free-living conditions, we need a reference in the pictures to account for the viewpoint and distance of the camera. For this purpose, the subjects are asked to place a reference card next to their food before taking a picture. The reference card is a standard 85.60x53.98mm [ISO/IEC 7813] ID card with a specific pattern printed on top. The pattern consists of two concentric rectangle (bull's-eye) patterns and a surrounding rectangle, as seen in Figure 3.1(a). The bull's-eye patterns are used to locate the reference card within the picture; and the surrounding rectangle is utilized for determining the four corners of the card.

The first step in reference card detection is to binarize the input image. Since global thresholding is likely to fail in capturing the local structures (therefore, the bull's-eye pattern), we employ an adaptive thresholding method [9]. In the adaptive thresholding, a gray scale image is taken as input, and then a threshold is computed for each pixel. Two major approaches are employed for calculating threshold, namely *Chow and Kaneko* and *local* thresholding. In the *Chow and Kaneko* approach, the image is split into subimages; and for each subimage, an optimum threshold is determined by examining its histogram. For each pixel, threshold is obtained by interpolating results of the subimages. On the other hand, the underlying idea behind of *local* thresholding is to utilize statistical functions, namely, mean and median. The intensity of local neighbor of each pixel is checked to use these functions. While employing the adaptive thresholding, we take the difference between the luminance channel of an image and its filtered version (which is obtained using an 11x11 averaging filter), and threshold the difference image to obtain the binary image. The pattern detector starts with the first row of the binary image, runs along the rows, first from left to right and then from right to left, and returns a high value at

center locations of alternating color patterns with mirror symmetry. Such bull's-eye detection is also used in localization of 2D barcodes, such as the MaxiCode [31]. Figure 3.1 illustrates that the adaptive thresholding method successfully extracts the local texture even if there is nonuniform illumination, and the reference card detector works well regardless of the orientation or the perspective of the card as seen in Figure 3.2. Once the patterns are located, we do a morphological region fill operation on the binary image to determine the exact location of the entire reference card. The seed points of the region fill operation are chosen as the points along the line that connects the centroids of the two peak regions (bull'seye centers) inside the card; this guarantees filling of the entire card region. After the entire card region is filled and the morphological operations are applied to the image, then the image is smoothed by a Gaussian filter to reduce noise. The Harris corner detector [32] is applied to the smoothed image to locate the four corners of the card, which can later be used for perspective correction of the food area estimates. The Harris corner detector is built on autocorrelation matrix *M*, whose formula is written explicitly below, I_x and I_y are the horizontal and vertical gradients of the image *I*, and *w(x,y)* is a weighting function, which is typically a Gaussian. The autocorrelation matrix at a pixel location is

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x(x,y)^2 & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y(x,y)^2 \end{bmatrix} \tag{3.1}$$

At a particular point, both of the eigenvalues of *M* matrix are large if the point is a corner. If the point is an edge point, one of the eigenvalues is large. If the point is on a smooth region, both eigenvalues are small. While cornerness response could be based on the eigenvalues, the Harris corner detector proposes a cornerness response that is based on the determinant and trace of the autocorrelation matrix;

$$R = \det M - k(traceM)^2 \tag{3.2}$$

where k is a small constant. After computing cornerness response of all points, non-maxima suppression is utilized to get the corner points. The cornerness response of the Harris corner detector is shown in Figure 3.3. Non-maximum suppression is then applied to locate the corners of the card.

3.2. Geometric Correction

In this section, we summarize the geometric transformation models [33], and we use to do geometric correction. A geometric transformation T maps the coordinates (x, y) in one image to the corresponding coordinates (x', y') in the other:

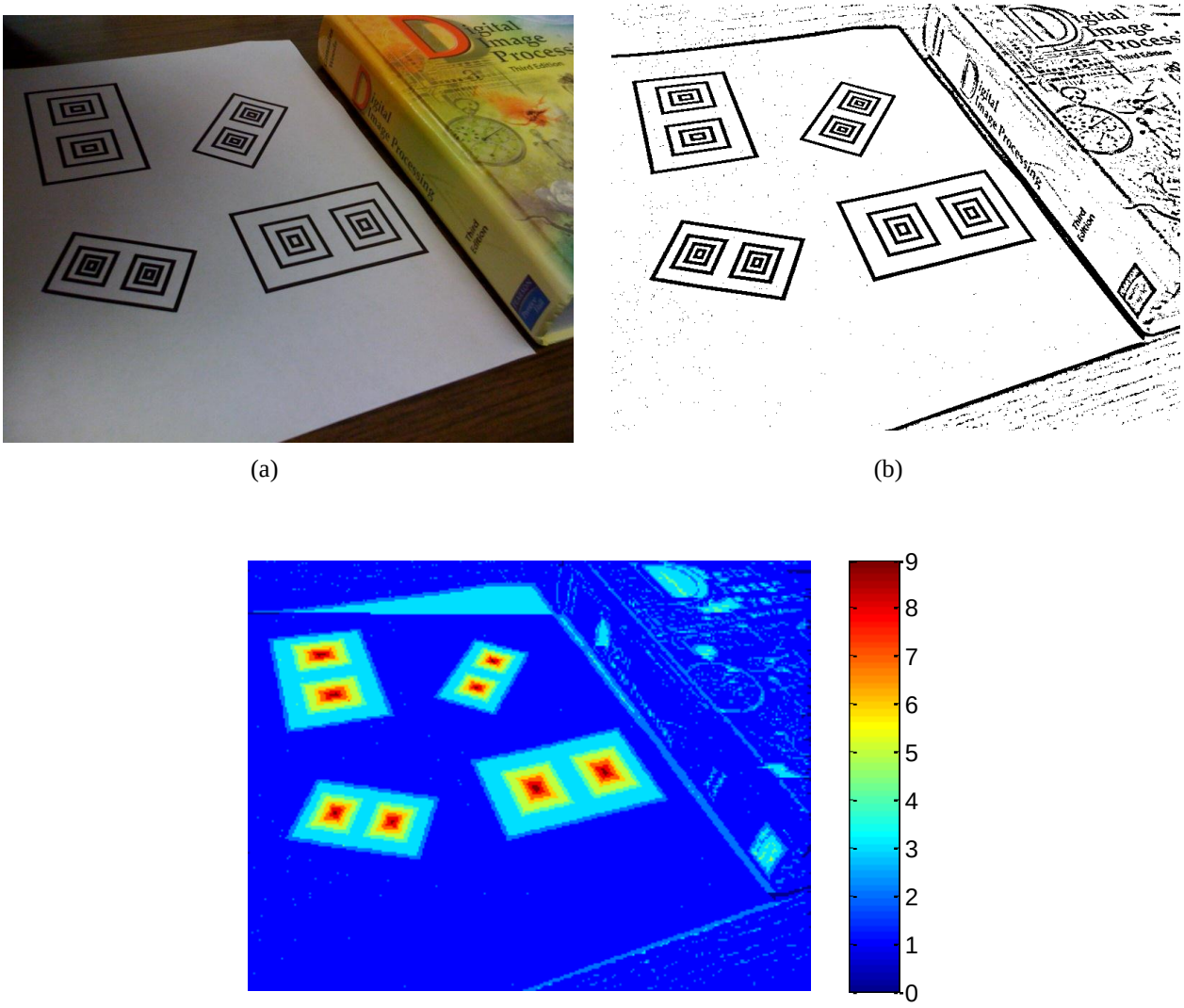
$$(x', y') = T(x, y) \tag{3.3}$$

To transform a *target image* on the coordinate system of a *reference image*, the transformation from target image to the reference image needs to be estimated. (See Fig. 3.4.) This can be achieved by fitting the model to a set of correspondences. Once the model parameters are estimated, we can obtain the corresponding coordinate for an arbitrary input coordinate. The corresponding coordinate will typically be a non-integer location; in that case, the nearest neighbors are used to interpolate the intensity at that location. Scanning the entire set of reference image coordinates, the target image is transformed onto the reference image coordinate system. In order to apply perspective or affine transformation to the input image, we need to have at least four or three corner pairs respectively. The reference card detector provides us four corners, and using these corners we can form a rectangle, which is proportional to size of the reference card, on the canvas image. Utilizing perspective or affine transformation models, the transformation coefficients are obtained. Then each pixel is transferred onto canvas image by using the transformation coefficients.

(a)

(b)

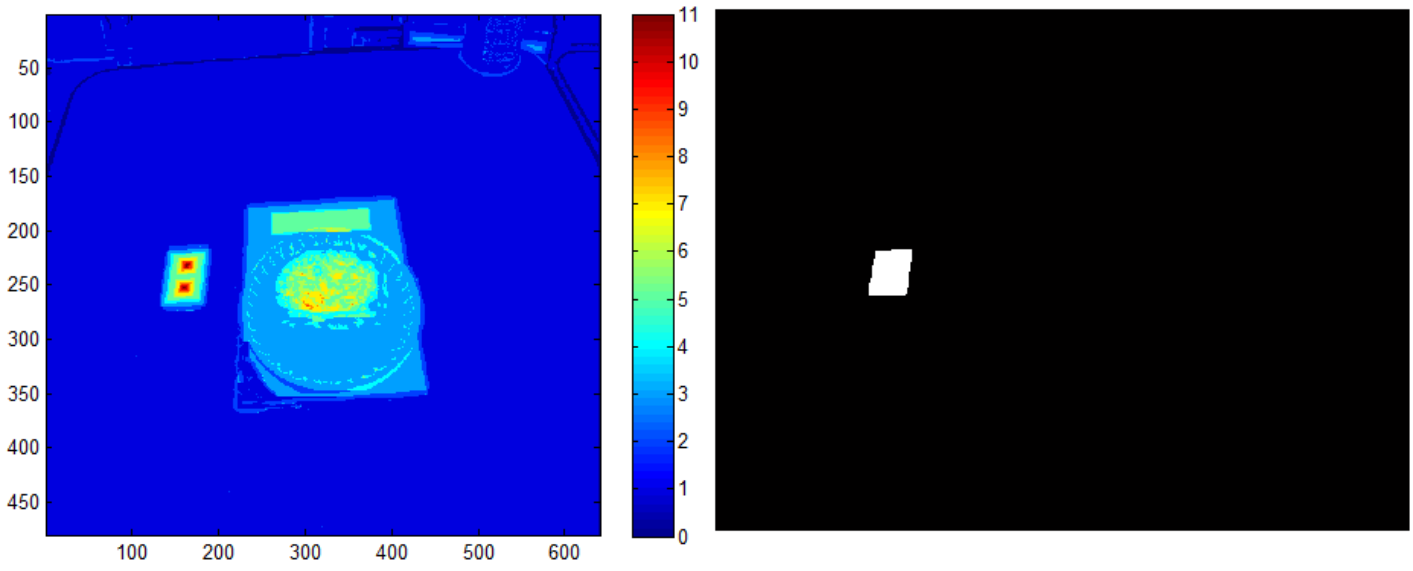
Figure 3.1: Adaptive thresholding for reference card detection: (a) Input image (b) Result of the adaptive thresholding.



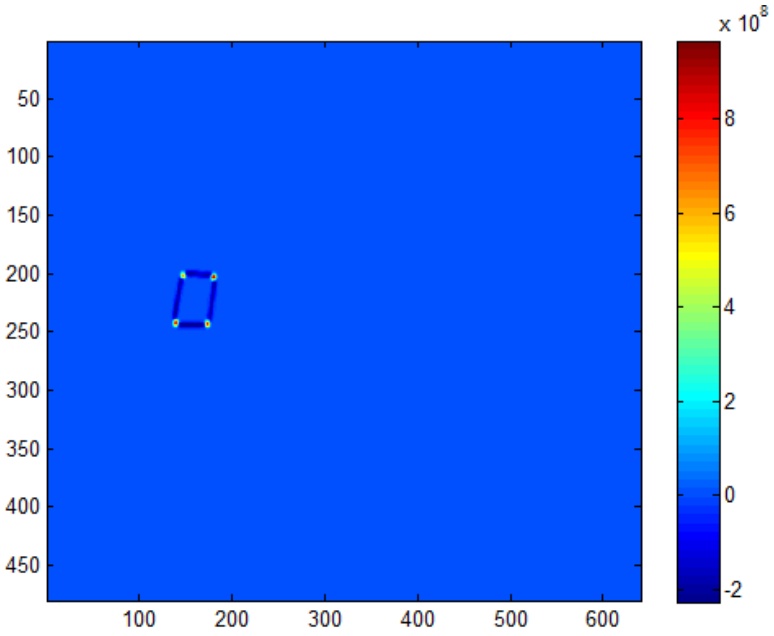
(a) (b)

(c)

Figure 3.2: Adaptive thresholding and reference card detection under various illumination conditions and geometric transformations: (a) Input image (b) Result of the adaptive thresholding (c) Response of the pattern detector.



(a) (b)



(c)

Figure 3.3: Reference card corner detection: (a) Response of the pattern detector applied to the binary image shown in Figure 3.1(b) (b) Detected reference card (c) Cornerness response of the Harris corner detector.

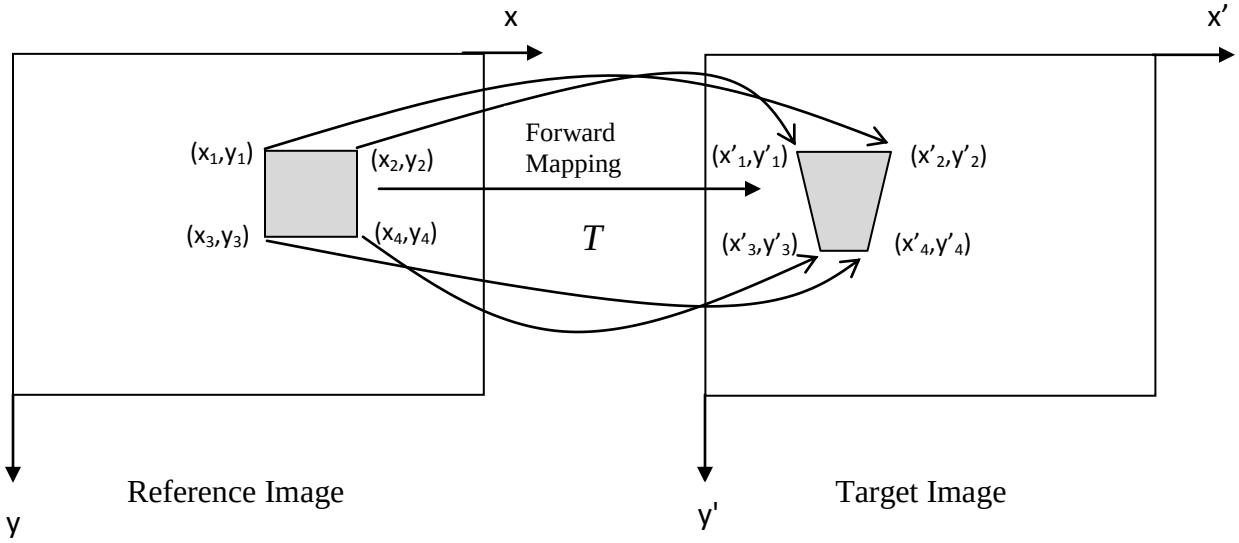


Figure 3.4: Forward mapping for spatial transformation of images.

3.2.1. Affine Transformation

The affine transformation model has six parameters and maps one coordinate system to another as follows [33, 34].

$$\begin{aligned} x' &= a_1 x + a_2 y + a_3 \\ y' &= a_4 x + a_5 y + a_6 \end{aligned} \tag{3.4}$$

The mapping can also be written as a matrix form as follows.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_3 \\ a_6 \end{bmatrix} \tag{3.5}$$

When we have a set of corresponding points, $i = 1, 2, 3, ..., N$ between *reference* and *target image*, the affine transformation coefficients can be obtained from these correspondences. *N* correspondences produce *2N* equations, and the value of *N* has to be at least 3 to calculate the 6 unknown parameters. This system can be expressed in a matrix form as follows.

$$\underbrace{\begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \end{bmatrix}}_{b} = \underbrace{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix}}_{a} \tag{3.6}$$

This matrix equation can be represented in a short form as $\boldsymbol{b} = \boldsymbol{A}\,\boldsymbol{a}$. The solution of this equation can be obtained using pseudo inverse:

$$\begin{aligned} A^T b &= A^T A a \\ a &= (A^T A)^{-1} A^T b \end{aligned} \tag{3.7}$$

Several common spatial transformations (scaling, translation, rotation and skew as illustrated in Figure 3.5) can be modeled by the affine transformation matrix. The limitation of the affine transformation is that it cannot map an arbitrary quadrilateral into another arbitrary quadrilateral, unlike the perspective transformation.

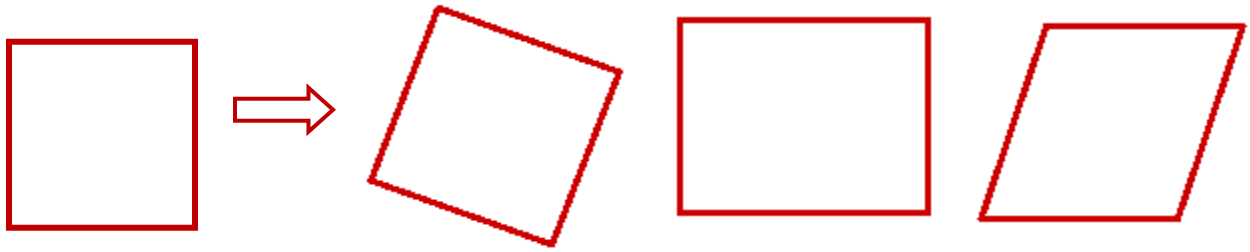


Figure 3.5: Transformations that can be generated by affine transformation.

3.2.2. Perspective Transformation

Perspective transformation model has eight parameters and is expressed as follows [33, 34].

$$x' = \frac{a_1 x + a_2 y + a_3}{a_7 x + a_8 y + 1} \qquad y' = \frac{a_4 x + a_5 y + a_6}{a_7 x + a_8 y + 1} \tag{3.8}$$

Since the perspective transformation has 8 coefficients in the transform matrix, this gives 8 degree of freedom to the perspective transformation, which is superior to the affine transformation. It can be also written in the form of homogeneous coordinates:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.9}$$

When we have a set of corresponding points, $i = 1, 2, 3, ..., N$ between *reference and target image*, perspective transformation coefficients can be obtained from this set of points. *N* points generate *2N* equations, in case of *N* equals 4, there are 8 unknown parameters. This system can be expressed in a matrix form as following.

$$\underbrace{\begin{bmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ . \\ . \\ x'_N \\ y'_N \end{bmatrix}}_{b} = \underbrace{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1 x'_1 & -y_1 x'_1 \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1 y'_1 & -y'_1 y'_1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2 x'_2 & -y_2 x'_2 \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2 y'_2 & -y'_2 y'_2 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ x_N & y_N & 1 & 0 & 0 & 0 & -x_N x'_N & -y_N x'_N \\ 0 & 0 & 0 & x_N & y_N & 1 & -x_N y'_N & -y'_N y'_N \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix}}_{a} \tag{3.10}$$

The given matrix equation can be represented as the following equation $\boldsymbol{b} = \boldsymbol{A}\,\boldsymbol{a}$. The solution of this equation can be obtained using pseudo inverse as in equation (3.7).

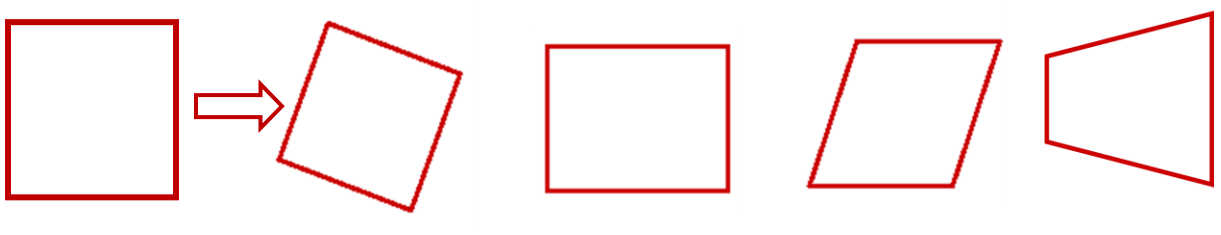


Figure 3.6: Transformations can be generated by perspective transformation.

3.2.3. Scaling

Scaling [33, 34] is such a transformation where the image size gets bigger or smaller. The idea behind of scaling is to change the number of pixels in the image contains by using interpolation techniques. This transformation is expressed with the following equations.

$$\begin{aligned} x' &= s_x x \\ y' &= s_y y \end{aligned} \tag{3.11}$$

This can be also written in a matrix form as follows.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{3.12}$$

In this study, we scaled the input images such that the number of pixels in the reference card area of the input image is equal to the number of pixels in the reference card area of the reference image. The horizontal and vertical scale factors (s_x,s_y) are kept equal; and nearest neighbor interpolation [9] is used to obtain the scaled image. An example of scale correction is depicted in Figure 3.7. In Figure 3.8, affine and perspective corrections are shown.

3.3. Photometric Correction

The purpose of photometric correction is to compensate when RBG pixel values are different from true values. The basic strategy of our approach is that the white regions of the

reference card are used to do white balancing, which improves the performance. The first step in this strategy is that we extract reference card region for each channel separately by using the mask of the reference card, as shown in Figure 3.3(b). After having extracted reference card region in the image (which is depicted Figure 3.9(c)), we find out maximum pixel value, and replace it with a proper ratio for each channel. So that maximum pixel value is set to 255 in each channel. This proper ratio becomes a correction coefficient for a particular channel. Then, for each pixel in each channel is scaled with its own coefficient to compensate, as illustrated in Figure 3.9.

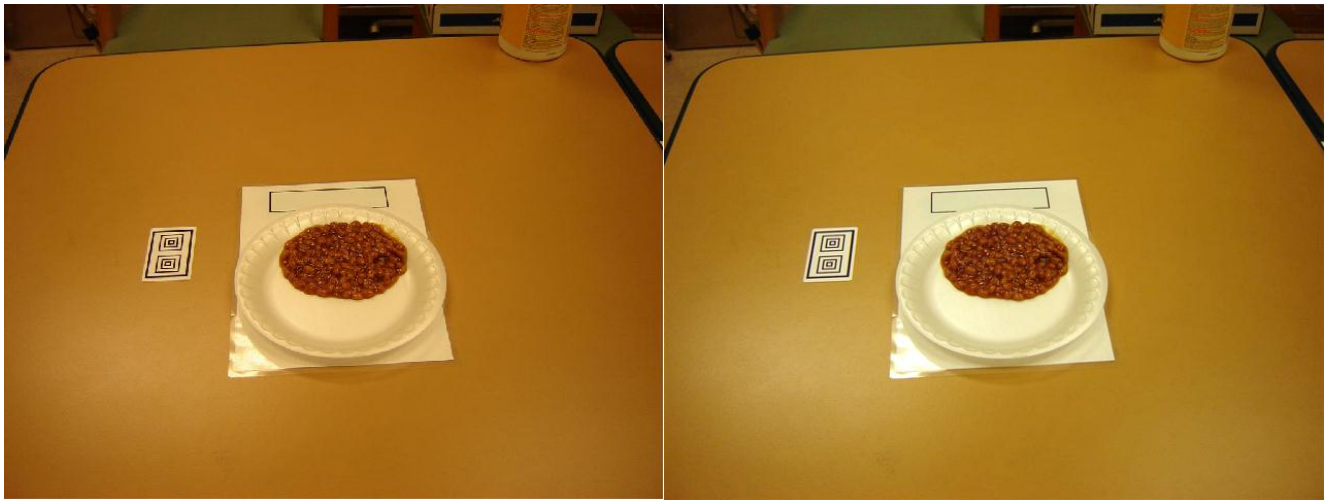
(a) (b)

Figure 3.7: Scale correction: (a) Reference image
(b) Scaled target image, such that the reference cards have the same number of pixels.

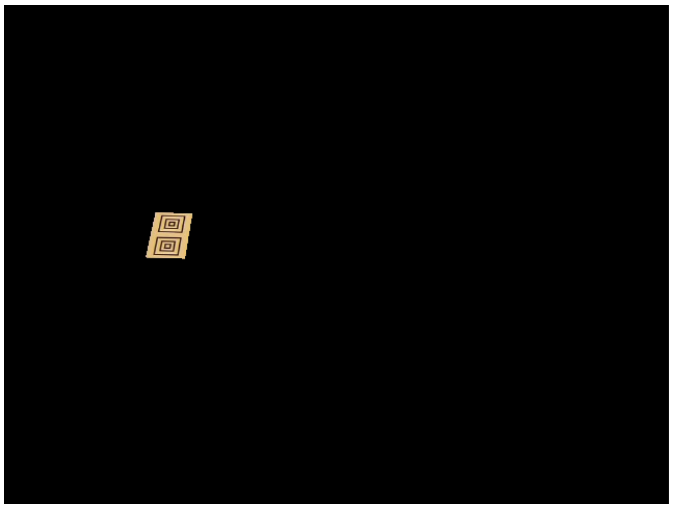
(a)

(b)

Figure 3.8: Affine and Perspective correction: (a) Affine transformed image (b) Perspective transformed image.



(a) (b)



(c)

Figure 3.9: Photometric correction: (a) Input image (b) Image after photometric correction (c) Extracted reference card region.

4. IMAGE ANALYSIS

4.1. Mahalanobis Distance

Mahalanobis distance [7, 34] is a distance measure which determines the similarity of an unknown set from a known sample set. Mahalanobis distance is also an extension of the Euclidean distance; it incorporates the covariance of the features included in the feature vectors. Mahalanobis distance between vectors $\boldsymbol{f} = [f_1, f_2, \cdots f_N]^T$ and $\boldsymbol{\mu} = [\mu_1, \mu_2, \cdots \mu_N]^T$ can be expressed as

$$d(\boldsymbol{f}, \boldsymbol{\mu}) = \sqrt{(\boldsymbol{f} - \boldsymbol{\mu})^T \mathbf{C}^{-1} (\boldsymbol{f} - \boldsymbol{\mu})} \tag{4.1}$$

where **C** is the covariance matrix of $\boldsymbol{f}$ and $\boldsymbol{\mu}$.

Mahalanobis distance indicates correlation between different features shows the scaling of axes. These capabilities make Mahalanobis distance superior to Euclidean distance. On the other hand, these attributes require more time and memory requirements.

Given a set of training data $\boldsymbol{f_i} = [f_{i1}, f_{i2}, ... f_{iN}]^T$, the mean and covariance matrix of the set can be estimated as [7, 34];

$$\boldsymbol{\mu} = \frac{1}{k} \sum_{i=1}^{k} \boldsymbol{f}_i \tag{4.2}$$

$$\boldsymbol{C} = \frac{1}{k} \sum_{i=1}^{k} (\boldsymbol{f}_i - \boldsymbol{\mu})(\boldsymbol{f}_i - \boldsymbol{\mu})^T \tag{4.3}$$

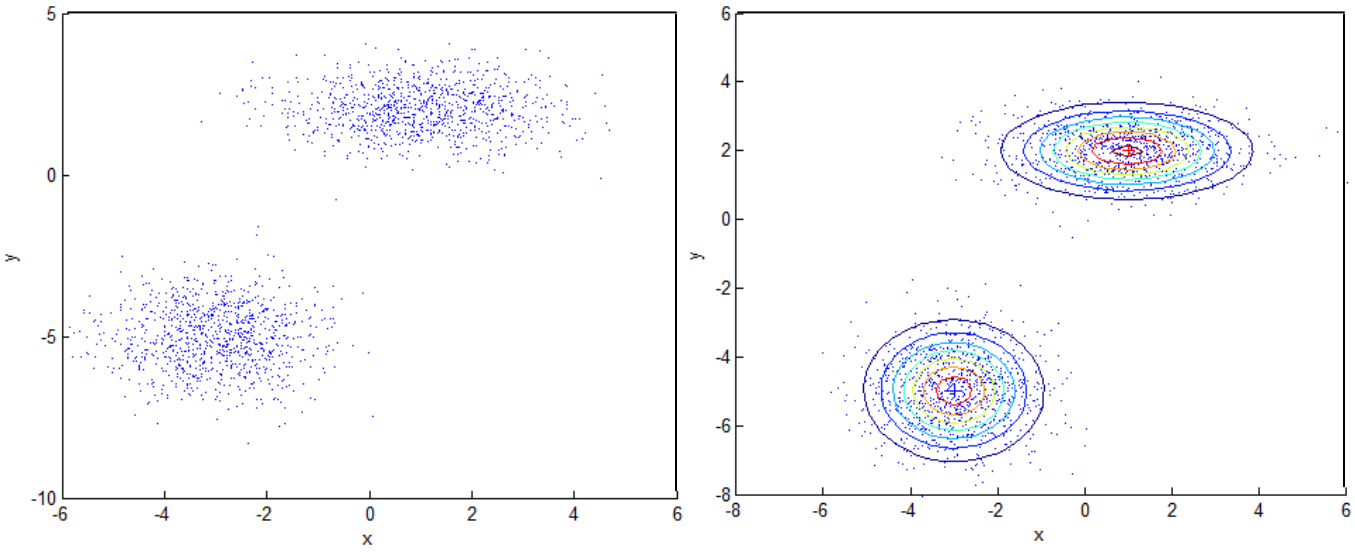
If the covariance matrix, **C**, is diagonal with equal values along the diagonal, the Mahalanobis distance reduces to Euclidean distance which is expressed as follows.

$$d(\boldsymbol{f}, \boldsymbol{\mu}) = \sqrt{(\boldsymbol{f} - \boldsymbol{\mu})^T (\boldsymbol{f} - \boldsymbol{\mu})} \tag{4.4}$$

Mahalanobis distance can be also used as a minimum-distance classifier. Given the mean and the corresponding covariance matrix of each class, a feature vector $\boldsymbol{f}$ can be classified by

determining the Mahalanobis distance from ***f*** to each of means, and assigning ***f*** to the class where Mahalanobis distance is minimum.

In classification, Mahalanobis distance (illustrated in Figure 4.1) can be computed to classify a testing point. After computing the covariance matrix for each class; for a given sample, the Mahalanobis distance is calculated to each class. The test sample point is assigned to the class where the Mahalanobis distance is the smallest.



(a) (b)

Figure 4.1: Sample figure for Mahalanobis distance: Center of each cluster is determined by the mean vector, and its shape of the cluster is determined by the covariance matrix.
(a) Distribution of two different classes (b) The ellipses show equal probability lines.

4.2. Multilayer Neural Networks and the Backpropagation Algorithm

4.2.1. Introduction

In a typical classification problem, linear discriminates are not capable of minimizing error issues where non-linearly separable patterns exist. On the other hand, selecting proper nonlinear functions are not as easy as it seems. One of the applicable solution approaches can be

addressed with the guidance of training data, namely multilayer neural networks, which can learn nonlinearity from training data [24].

The fundamental of multilayer network [24] is based on the backpropagation algorithm. The backpropagation is also known as gradient descent in error. Training for multilayer network is much easier and applicable for even so complicated classification problems due to the fact that it can be analyzed graphically. The backpropagation is basically generalization of the LMS (Least-Mean-Squares) algorithm. The idea of the backpropagation is to increase training speed, improve performance and obtain desired output values by adjusting weights and scaling inputs. Selecting the neural network structure is extremely important in classification problems. This may also cause overfitting or underfitting for the training samples. Adjusting complexity of neural network architecture is known as regularization.

The single perceptron can only be used for the classes that are linearly separable whereas a typical multilayer network [35], which is learned by the backpropagation algorithm, has aptitude to classify the classes which are not necessarily linearly separable.

4.2.2. Feedforward Operation

A simple three-layer neural network [24], shown in Figure 4.2., has input, hidden, and output layers. There is also a single bias unit that has connection with all units except input units. Since these connected components treat like biological neurons, they are named as neurons. All input vectors are presented to the input layer, and the outputs of input neurons corresponds to component vectors. Net activation is computed in hidden neurons which take weighted sum of inputs into account. Briefly, net activation is the sum of the inner products of weight and input vectors added by initial weight. The net activation is shown below

$$net_j = \sum_{i=1}^{d} x_i w_{ji} + w_{j0} \tag{4.5}$$

where *i* denotes *ith* input, and *j* denotes a node (neuron) in the hidden layers. The activation function $f(\cdot)$ produces output of the hidden neuron:

$$y_j = f(net_j) \tag{4.6}$$

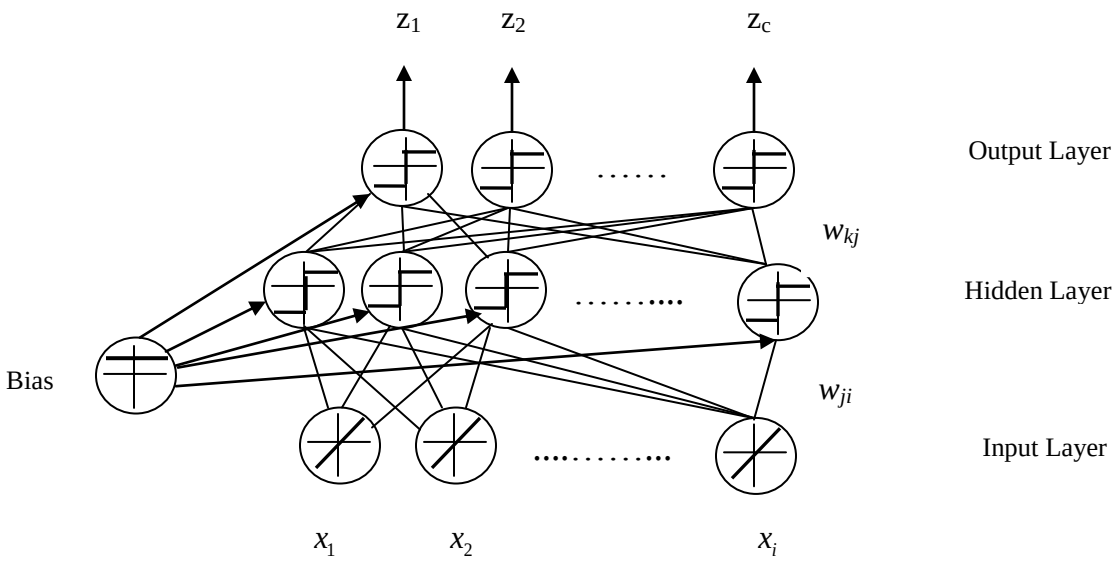


Figure 4.2: A simple three-layer neural network with bias.

As the expression (4.7) is shown below, net activation is calculated for each output neuron in the hidden neurons.

$$net_k = \sum_{j=1}^{n_H} y_j w_{kj} + w_{k0} \tag{4.7}$$

where n_H is the number of hidden neurons, and w_{kj} is the weight from *jth* hidden neuron to *kth* output neuron. The output of the *kth* neuron is as follows.

$$z_k = f(net_k) \tag{4.8}$$

Since the multilayer neural network has ability to use hidden layer and various functions (such as linear, sign and etc.), this feature makes multilayer network [24] more powerful compare to other networks. The equations above, (4.5) through (4.8), can be generalized for the multilayer neural

network which has c output neurons and whose discriminant function $g_k(x)$, the signal from each output neurons, as follows.

$$g_k(x) \equiv z_k = f\left(\sum_{j=1}^{n_H} w_{kj} f\left(\sum_{i=1}^{d} w_{ji} x_i + w_{j0}\right) + w_{k0}\right) \tag{4.9}$$

4.2.3. The Backpropagation Algorithm

The Backpropagation algorithm [24] is one of the most common methods in multilayer neural networks for supervised learning. The major issue here is to obtain desired outputs adjusting weight vectors based on inputs provided by training sets. The idea behind the backpropagation algorithm is to obtain the sum of the smallest squared difference between actual outputs and desired outputs. This learning rule is valid for not only two-layer architecture but also three-layer architecture. The underlying idea behind network learning is to present a training pattern to an untrained network's input layer and to let the signal pass through hidden layer to output layer. Output layer then generates output. The difference between generated output and desired output (target value) is called an error. The aim of the learning rule, which is presented here, is to minimize the error adjusting the weights and presenting learning rule is also pattern basis method. The training error on a pattern basis can be defined as the sum of the squared difference between desired output (target value) t_k and the actual output z_k .This can be expressed with the following equation [24].

$$J(w) \equiv \frac{1}{2}\sum_{k=1}^{c}(t_k - z_k)^2 \tag{4.10}$$

In the network, the size of output vector is *c* and *w* is the vector consisting of all the weights. Here, the backpropagation is gradient descent algorithm which changes the weights values in a direction where the error is minimized. This is expressed as following:

$$\Delta w = -\eta \frac{\partial J}{\partial w} \tag{4.11}$$

where η, namely learning rate, controls how much change occurs in weights. With each update of (4.11), the criterion function $J(w)$ is to reduced. According to the learning rule, the learning will eventually stop at local or global minima. This iterative approach can be expressed as follows:

$$w(m+1) = w(m) + \Delta w(m) \tag{4.12}$$

where it takes a weight vector for the particular index *m* and updates it. In case of a three-layer net in the equation (4.11), the chain rule is used for differentiation assuming that w_{kj} is the hidden-to-output weights

$$\frac{\partial J}{\partial w_{kj}} = \frac{\partial J}{\partial net_k} \frac{\partial net_k}{\partial w_{kj}} = -\delta_k \frac{\partial net_k}{\partial w_{kj}} \tag{4.13}$$

where the equation (4.14) is called the sensitivity of unit *k* and that shows how the general error changes.

$$\delta_k = -\partial J \Big/ \partial net_k \tag{4.14}$$

When we differentiate equation (4.10), we can obtain such an output, δ_k as follows.

$$\delta_k = -\frac{\partial J}{\partial net_k} = -\frac{\partial J}{\partial z_k} \frac{\partial z_k}{\partial net_k} = (t_k - z_k) f'(net_k) \tag{4.15}$$

The last derivative in equation (4.13) can be generated as follows.

$$\frac{\partial net_k}{\partial w_{kj}} = y_j \tag{4.16}$$

These results provide us the weigh update or learning rule for the hidden-to-output weights as follows.

$$\Delta w_{kj} = \eta \delta_k y_j = \eta (t_k - z_k) f'(net_k) y_j \tag{4.17}$$

If the output is linear, then as the backpropagation rule becomes follows.

$$f'(net_k) = net_k \tag{4.18}$$

$$f'(net_k) = 1 \tag{4.19}$$

By using chain rule, from equation (4.11), we compute

$$\frac{\partial J}{\partial w_{ji}} = \frac{\partial J}{\partial y_j} \frac{\partial y_j}{\partial net_j} \frac{\partial net_j}{\partial w_{ji}} \tag{4.20}$$

The right side of the equal sign in the first term should be taken care of carefully:

$$\begin{aligned} \frac{\partial J}{\partial y_j} &= \frac{\partial J}{\partial y_j} \left[\frac{1}{2} \sum_{k=1}^{c} (t_k - z_k)^2 \right] \\ &= -\sum_{k=1}^{c} (t_k - z_k) \frac{\partial z_k}{\partial y_j} \\ &= -\sum_{k=1}^{c} (t_k - z_k) \frac{\partial z_k}{\partial net_j} \frac{\partial net_j}{\partial y_j} \\ &= -\sum_{k=1}^{c} (t_k - z_k) f'(net_k) w_{kj} \end{aligned} \tag{4.21}$$

The chain rule is applied to the second step as well. The final, in equation (4.21), step plays crucial role in order to calculate effective target activation for each hidden unit since it indicates that how the hidden output unit affects the error. For a hidden unit, the sensitivity can be described with using equation (4.21) as follows.

$$\delta_j \equiv f'(net_j) \sum_{k=1}^{c} w_{kj} \delta_k \tag{4.22}$$

Then the learning rule can be also expressed which consist of the sum of individual sensitivities for a hidden unit as follows.

$$\Delta w_{ji} = \eta x_i \delta_j = \eta \left[\sum_{k=1}^{c} w_{kj} \delta_k \right] f'(net_j) x_i \tag{4.23}$$

The equations (4.23) and (4.17) basically address the backpropagation algorithm, which is also called backpropagation of errors due to fact that while performing learning of the input-to hidden weights, the error must be occurred. A description of the backpropagation algorithm can be found in [24].

4.3. Training Stage

The goal of the training stage is basically to obtain essential variables and coefficients and to incorporate them into the testing stage. Training stage consists of reference card detection, region selection & feature extraction and training of the classifiers steps. The flow chart of the training stage is shown with details in the Figure 1.1. Since reference card detection step is already taken care of in the Section 3, here, we will explain region selection & feature extraction and training of the classifiers steps.

4.3.1. Region Selection and Feature Extraction

This section consists of region selection, in which the food region is outlined, and feature extraction of the food, in which R, G, B color and Gabor texture features [37, 38] are utilized, in the train image. This requires extraction of the features for each food type in a training process. During training stage, a user manually selects the food region for each food type. How the feature extraction process occurs is that the matrix multiplication, element by element, is performed between manually segmented food region as illustrated in Figure 4.4(b) and R, G, B channels which are individually extracted from the image. To extract Gabor texture features from

the train image, we will consider Gabor Filter. A two-dimensional Gabor function is composed of a modulated a sinusoid by a Gaussian along with some frequency and orientation. In special domain, a complex Gabor function can be expressed as follows [37, 38],

$$g(x, y) = s(x, y)w_r(x, y) \tag{4.24}$$

where *s(x,y)* is a complex sinusoid, and $w_r(x,y)$ is a Gaussian-shaped function, namely the envelope. The complex sinusoid is viewed as below:

$$s(x, y) = \exp(j(2(u_0 x + v_0 y + P)) \tag{4.25}$$

where (u_0, v_0) is the spatial frequency, and *P* is the phase of the sinusoid of the complex component, whereas the Gaussian envelope is expressed as follows:

$$w_r(x, y) = K \exp(-\pi(a^2(x - x_0)_r^{\ 2} + b^2(y - y_0)_r^{\ 2})) \tag{4.26}$$

where (x_0, y_0) is the peak location, *K* is the scales the magnitude, (a, b) is the scaling parameters of the Gaussian envelope, and *r* is the rotation operation.

The Gabor texture features are generated as follows. At each pixel, we do following transformation:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{4.27}$$

Then, Gabor filter is applied to Y channel as follows:

$$G_i(x,y) = Y(x,y) * g_i(x,y) \tag{4.28}$$

where *g(x,y)* is a Gabor filter. After having filtered image, element by element matrix multiplication is performed between the filtered image and the segmented image, which is consists of a matrix. This operation yields the Gabor texture feature vectors. In Figure 4.3, the Gabor filter response and filtered image sequences are shown with parameters' various values.

Then the features associated with that food type are extracted and saved. (Figure 4.3. and 4.4. show a screenshot of the region selection and feature extraction process.) The user is also asked to enter the gram amount (or volume) of the food type; this information is used to establish the association between the food region area and the gram amount for each food type. In our prototype system, we use the color RGB data (red, green, and blue values) and Gabor texture as the feature vector for the classifier as follows.

$$f(x,y) = \begin{bmatrix} R(x,y) \\ G(x,y) \\ B(x,y) \\ G_1(x,y) \\ G_2(x,y) \\ G_3(x,y) \\ G_4(x,y) \\ G_5(x,y) \\ G_6(x,y) \end{bmatrix} \tag{4.29}$$

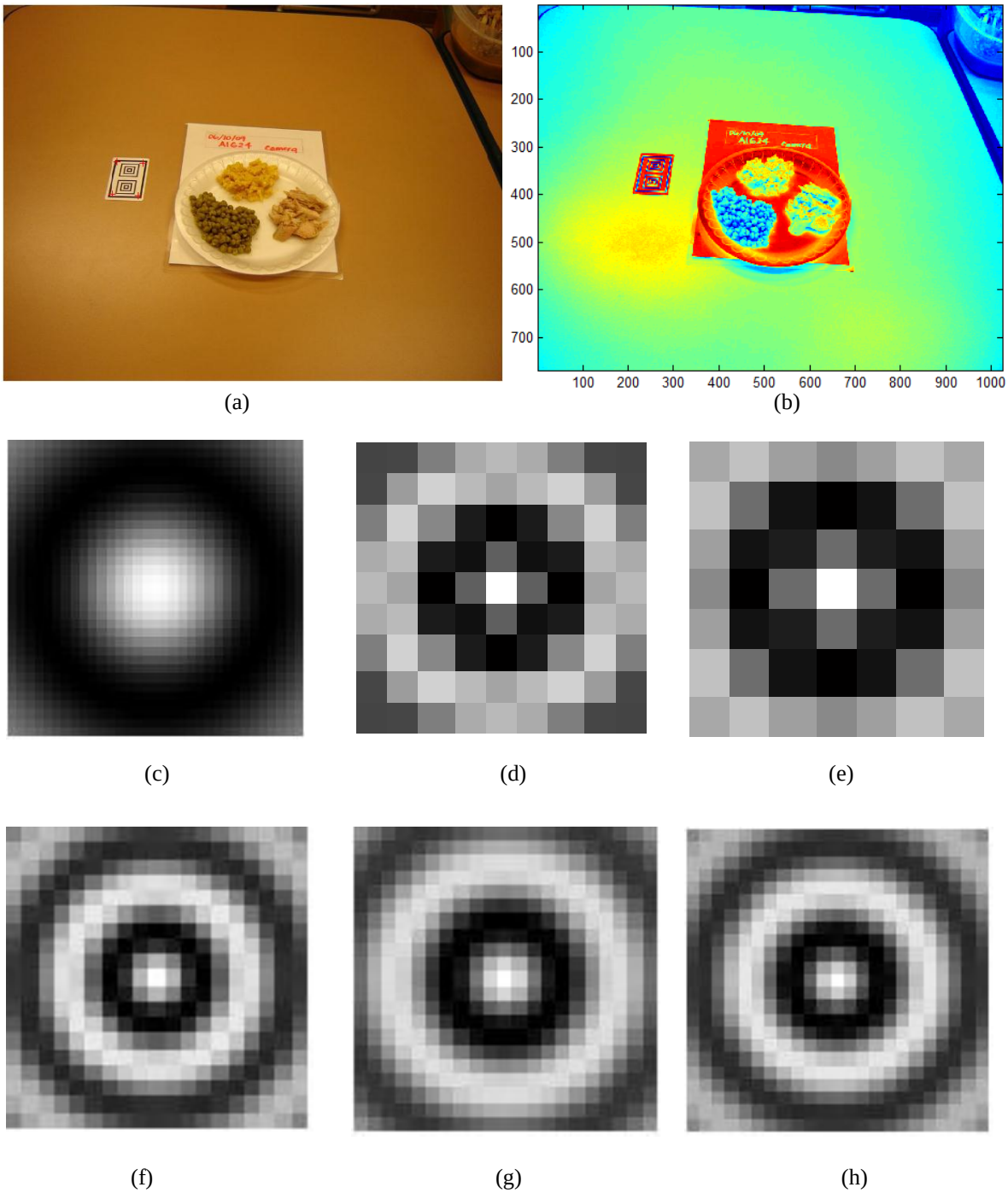
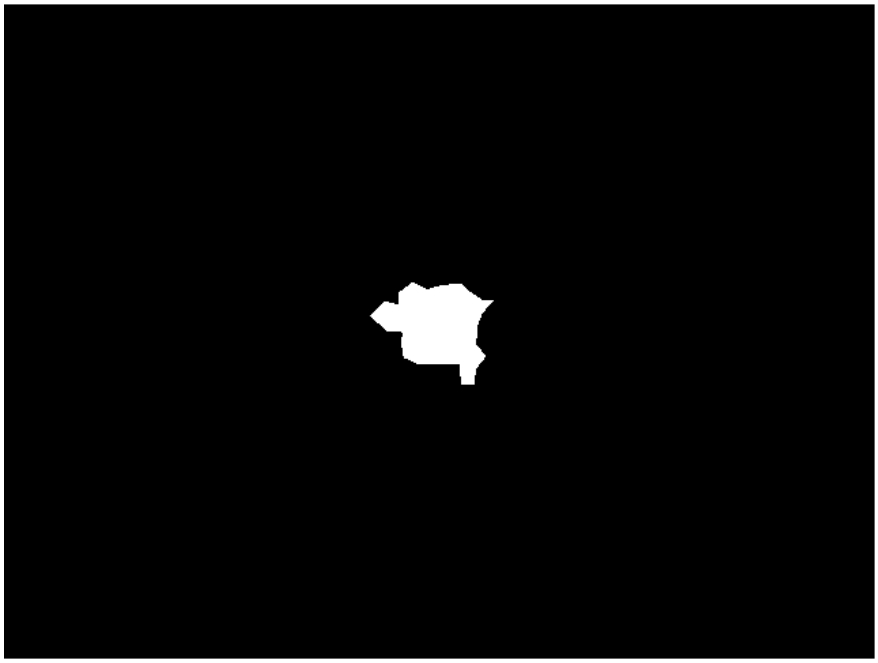


Figure 4.3: Gabor texture feature extraction: (a) Input image (b) Pseudo-colored luminance channel of the input image. The Gabor filters used to obtain texture features: (c) Scaling parameters a=b=16 $u_0 = v_0$ =0.01125π (d) a=b=4 $u_0 = v_0$ =0.08725π (e) a=b=3.5 $u_0 = v_0$ =0.083125π (f) a=b=9.5 $u_0 = v_0$ =0.05594π (g) a=b=10.5 $u_0 = v_0$ =0.040625π (h) a=b=11 $u_0 = v_0$ =0.0453125π.

(a)



(b)

Figure 4.4: Manual selection of the food region during training stage:
(a) The region of interest is outlined by the blue points that are clicked by the user
(b) Segmented the region of interest to extract features from the food region.

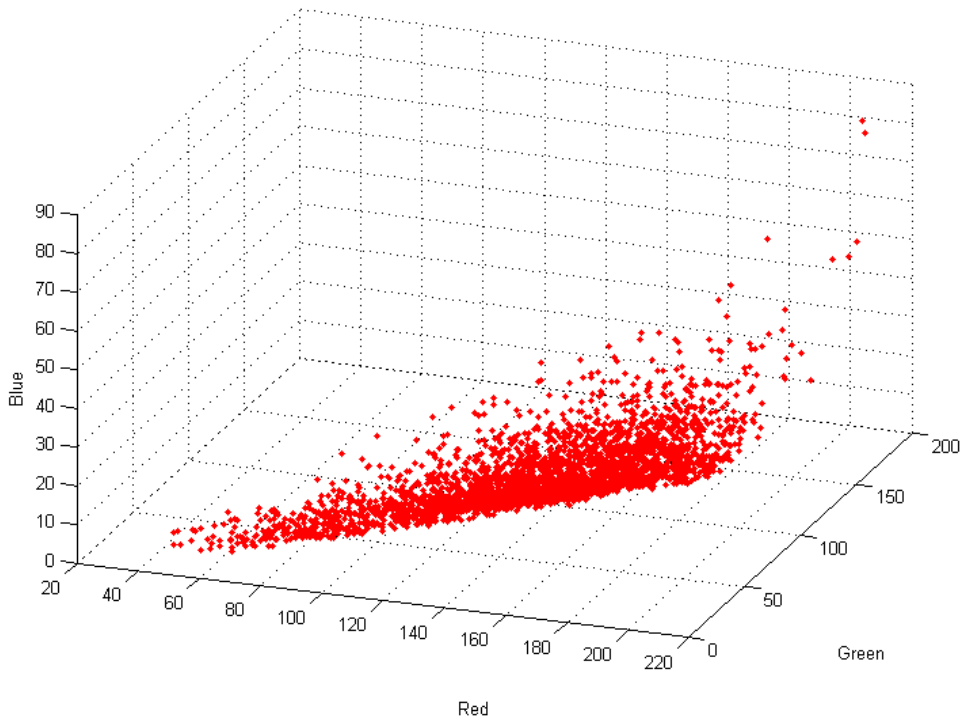


Figure 4.5: Distribution of the color samples in RGB space for the region selected in Figure 4.4.

4.3.2. Training of the Classifiers

4.3.2.1. Training of the Mahalanobis Distance

The distribution of the RGB data in a selected food region is modeled as a Gaussian (as seen in Figure 4.5); the parameters of the Gaussian model are mean and covariance matrix. During the training stage, the mean and covariance matrix are computed for a particular food type using extracted features from the food region in the image. (We should also note here that the white regions of the reference card are used to do white balancing, which improves the performance.) If the features are not extracted yet, the features will extracted from the food region in the image before computing mean and covariance matrix. Once these operations are performed, essential coefficients (computed mean and covariance, food type, feature type) are saved for each food type to be used in the testing stage.

4.3.2.2. Training of the Neural Network

In general, training of a feedforward neural network consists of four major steps [36] as follows: obtaining training data, generating the network object, training the network and simulating the network response to test inputs. According to these steps, the first thing we need to do is to assemble the training data which are basically feature vectors and target vectors for each food type. The feature vectors are already obtained during the region selection and feature extraction section, whereas target vectors, which correspond to particular food type, are needed to be generated. The idea behind of generating target vectors is that 1 is assigned to the corresponding class and others become 0. With this procedure, target vectors are generated for all classes.

Once the feature vectors and target vectors are available to be used, the first step of training the feedforward network, which is to create a network object, can be implemented. In Matlab, the *newff* function creates the feedforward network structure which requires at least three input arguments and returns the network object. The first argument is a matrix which consists of feature vectors whose length is number of feature types by the length of segmented region. The second argument is a matrix which contains target vectors whose length is number of food types (food classes) by the length of segmented region. The third argument is an array which contains size of each hidden layer in the feedforward network. There are several optional arguments for which can be used in the *newff* function. In the proposed system, the default training functions for hidden layers *tan-sigmoid*, and for the output layer is linear. Besides, the default training functions, *trainlm*, is used in the *newff* due to fact that it is very fast. The number of epochs and the number neurons in the hidden layer are set to 20 for the existing system.

After creating the network object, the next step for training the feedforward network is to initialize weights and biases even though *newff* command initializes the weights. This is done

with init command. This function takes network object as input and returns it with all weights and biases initialized.

The next step after having initialize weights is to simulate network which takes feature vectors and the network object net as inputs and returns the network outputs. In our proposed system, we do simulation before and after training to be able to compute mean-squared error for both cases.

After completing these steps; creating the network object, initializing weighs and simulating networks, the next important step is to train the feedforward network. During training (as seen in Figure 4.6 and 4.7), weights and biases of the network are iteratively adjusted to minimize the network performance function, namely MSE, the average squared error difference between the target vectors and the network outputs. During training, Figure 4.6 interprets how mean squared error is getting decreased from a large value to a smaller value which is simply called learning of the network. The plot is divided into 3 lines which correspond to three sets; 60% of the set is used for training, 20% of the set is used for validation, and the last 20% of the set become a test set for the network.

The backpropagation technique is used to determine the gradient of the performance which adjusts the weights to minimize error. Since the gradient descent algorithm works pretty slower compare to the several backpropagation algorithms in practical problems, in our system we use the Levenberg-Marquardt [36] backpropagation algorithm (as seen in Figure 4.7) which is based on numerical optimization technique.

4.4. Testing Stage

The goal of the testing stage is to determine and obtain the gram amounts of food classes by using either the Neural Network [36] or the Mahalanobis Distance [24] classifier and utilizing the segmentation and gram estimation algorithms, as well as the reference gram and surface area

data extracted from training stag. Testing stage consists of reference card detection & geometric transformation, pixel classification, segmentation, and gram estimation steps which are illustrated in detail in Figure 1.2. Since reference card detection & geometric transformation step is already explained in Section 3, we will point out pixel classification, segmentation and gram estimation steps.

4.4.1. Pixel Classification

4.4.1.1. Pixel Classification with Mahalanobis Distance

In the proposed system, there are two types of image sequences, namely before intake and after intake images (as illustrated in Figure 4.8 and 4.9). During pixel classification stage by Mahalanobis distance [24] technique, the distance between each pixel and a food class is calculated using the Mahalanobis distance, which accounts for the distribution of the feature values. A typical distance image sequence is illustrated in Figure 4.8 and 4.9 for particular classes. The distance calculation process is repeated for all food classes. Pixels that are close to a class (i.e., pixels with Mahalanobis distance less than a pre-determined threshold) are assigned to that class. After repeating for all classes, we have the food regions for each class. After this process, the next step would be segmentation and post processing which is described in the next sections where food regions are segmented and classified.

4.4.1.2. Pixel Classification with Neural Network

This section briefly explains how to obtain the distance of pixels to a particular food type incorporating Neural Network [36] into the system. In order to begin with the distance calculation process, the first step is to extract and generate feature vectors, namely RGB color and Gabor texture features [37], from the test image. Since manually selection process does not exist in the testing stage, instead we generate ones matrix whose size is the same with the test image. Then to create color feature vectors, the ones matrix, which is basically segmented food

region for testing image, is applied to each channel, namely RGB, separately for performing element by element matrix multiplication. On the other hand, to generate the Gabor texture feature, firstly test image is converted to gray scale image which is become an input argument for the Gabor filter function. After having filtered image, element by element matrix multiplication is performed between filtered image and the ones matrix. This operation yields the Gabor texture feature vectors. From the training stage, what we have is trained net object which is used as one of the input arguments to simulate network. The other input argument is the generated feature vectors. The network is simulated to yield an output vector in which output neurons correspond to the particular food type. After the output vector is generated, then the Euclidean distance is computed as the sum of squared difference for each food type gives us the distance matrix for each food type. As Section 4.4.1.1, there are two image pattern sequences, namely before intake and after intake, which are depicted in Figure 4.10 and 4.11. The obtained distance images are shown in Figure 4.10 and 4.11 for selected classes.

4.4.2. Segmentation

This section consists of two major processes, namely segmentation and post processing. In the segmentation process, the essential element is the distance matrix which is yielded as an output after pixel classification with Mahanalonobis Distance [24] or Neural Network [36] section. The underlying idea of segmentation process is that the smallest element is chosen for each pixel among all food classes, and this operation is repeated for all pixels. Thus, this process produces an output image matrix which is composed of integers where each number represents a food class. Within segmentation process, the next step is basically to apply mode filter [9] to the output image matrix, as well as ignoring bad pixels that have no value. What mode filter does is that each pixel value is replaced by its most common neighbor. Once this operation is performed,

final resulted image can be obtained as follows. Before and after intake image patterns are illustrated in Figure 4.12 and 4.15.

The aim of the post processing is to obtain much accurate classification and segmentation for each food type by getting rid of pixels which are classified as a food type. The way which is followed is composed of three parts; removing tiny regions, clearing targeted regions and clearing the outside of defined rectangle boundary mask. The idea of removing tiny regions is that whichever region is smaller than certain threshold is removed (by assigning zero) from the classified image. Clearing targeted regions is based on defined boundary rectangle mask around the plate. Targeted regions, which touch to the boundary rectangle mask, are cleared with the same idea. Lastly, the outside of defined rectangle boundary is cleared. The classified and segmented image patterns are seen in Figure 4.12 through 4.15 during before and after post processing.

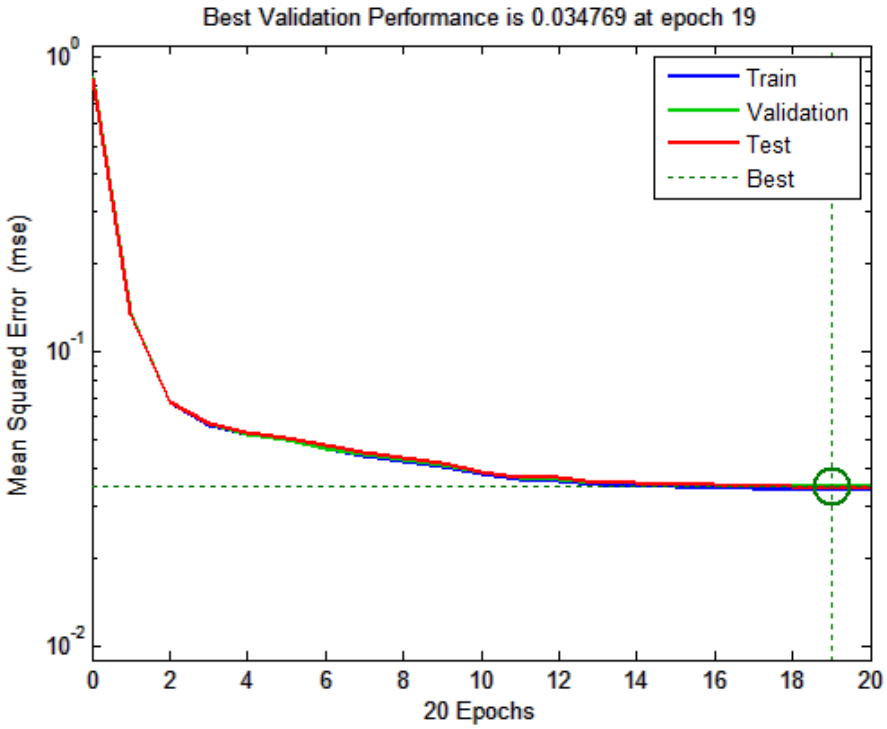


Figure 4.6: The change in MSE as a function of the number epochs is plotted

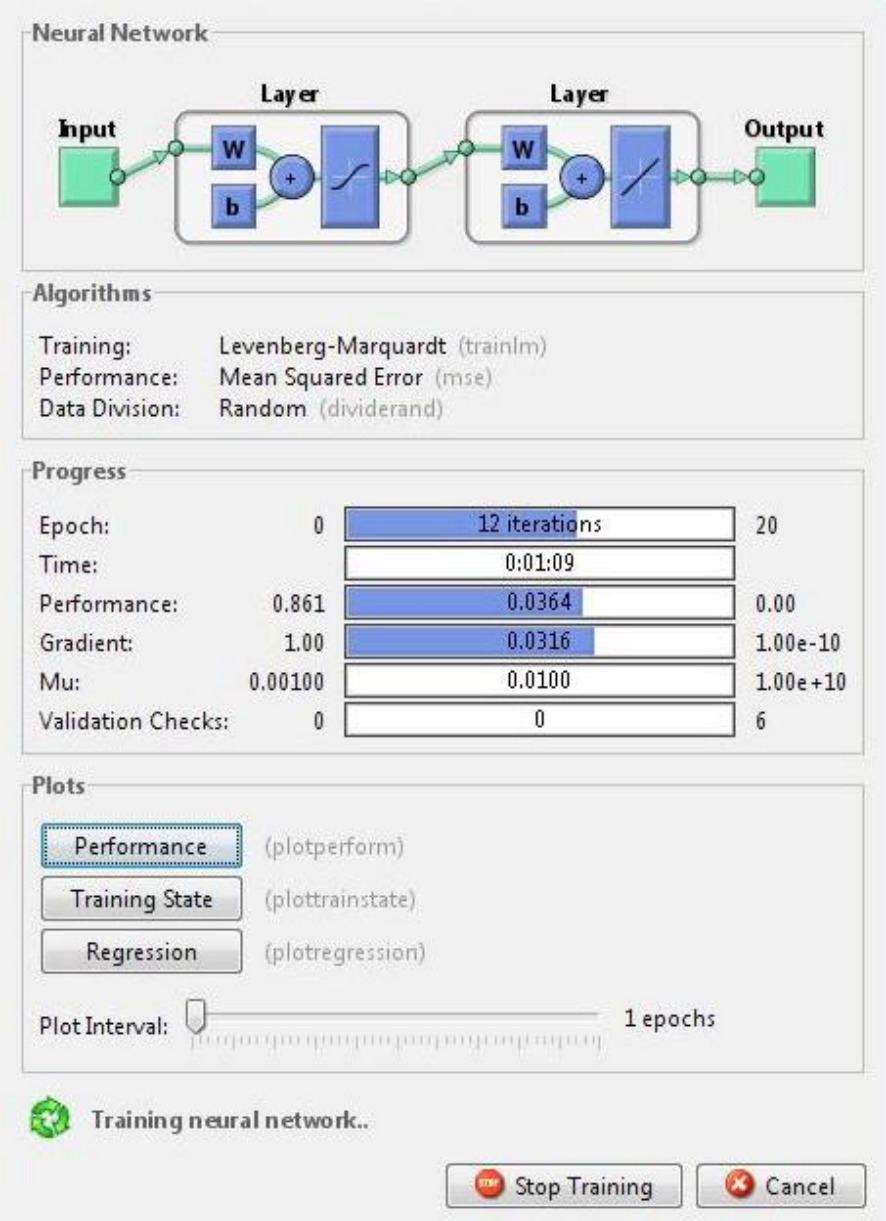


Figure 4.7: A screen shot of neural network training tool in Matlab

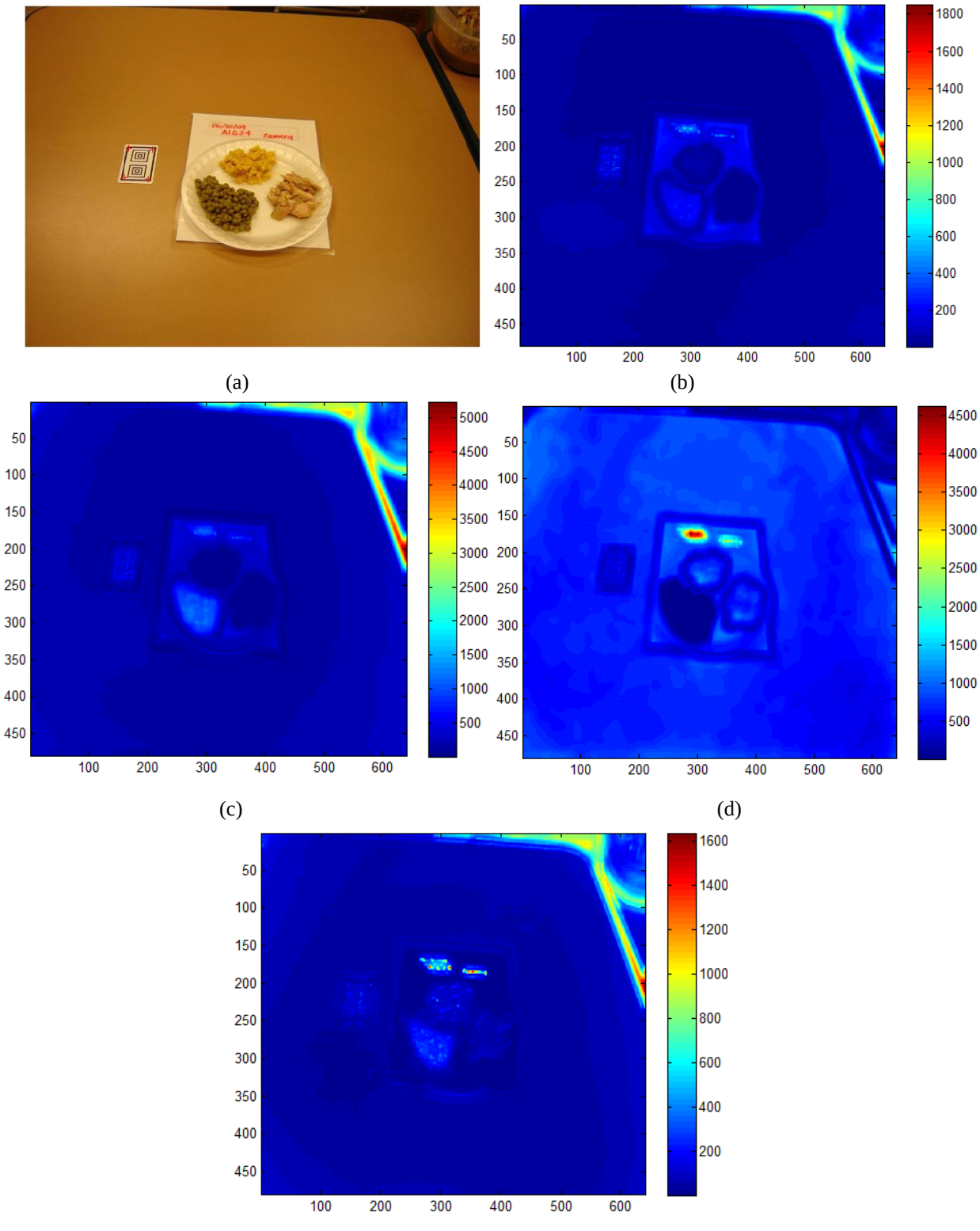


Figure 4.8: Region segmentation and pixel classification (before intake): (a) Input test image. Mahalanobis distance of pixels to the food class: (b) Chicken breast pieces (c) Macaroni and cheese (d) Sweet peas (e) Plate.

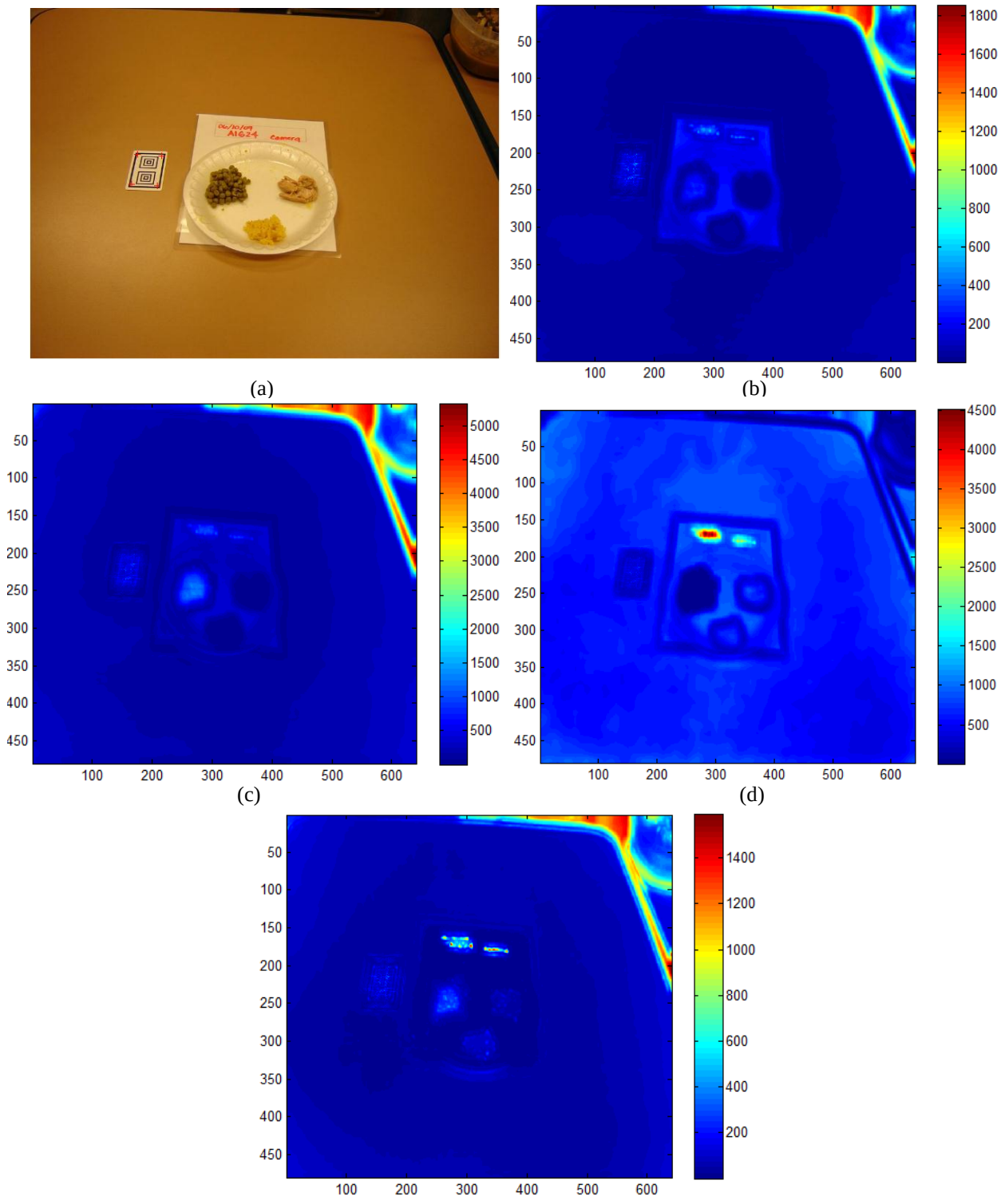


Figure 4.9: Region segmentation and pixel classification (after intake): (a) Input test image. Mahalanobis distance of pixels to the food class: (b) Chicken breast pieces (c) Macaroni and cheese (d) Sweet peas (e) Plate.

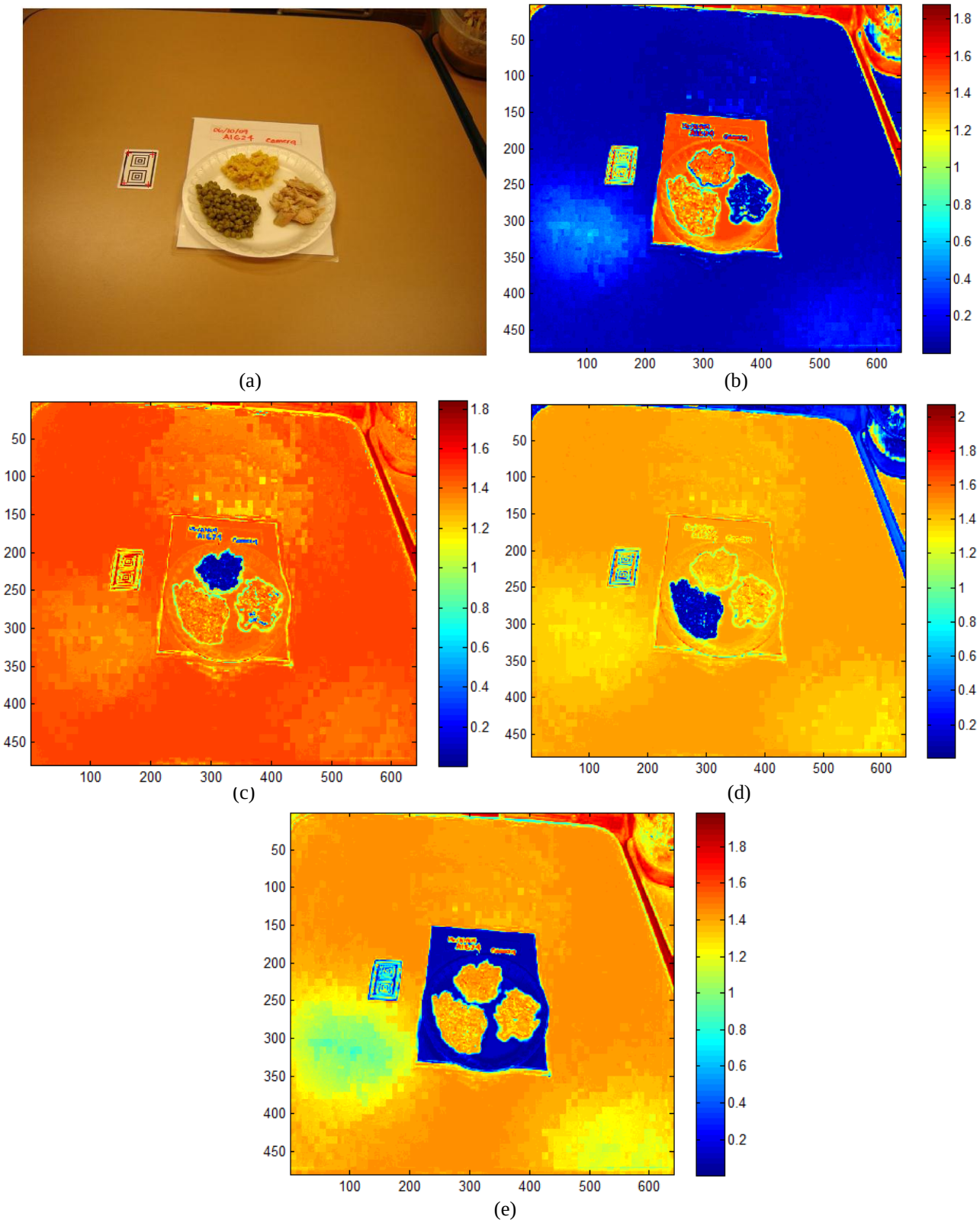


Figure 4.10: Region segmentation and pixel classification by Neural Network classifier (before intake): (a) Input test image (b) Chicken breast pieces (c) Macaroni and cheese (d) Sweet peas (e) Plate.

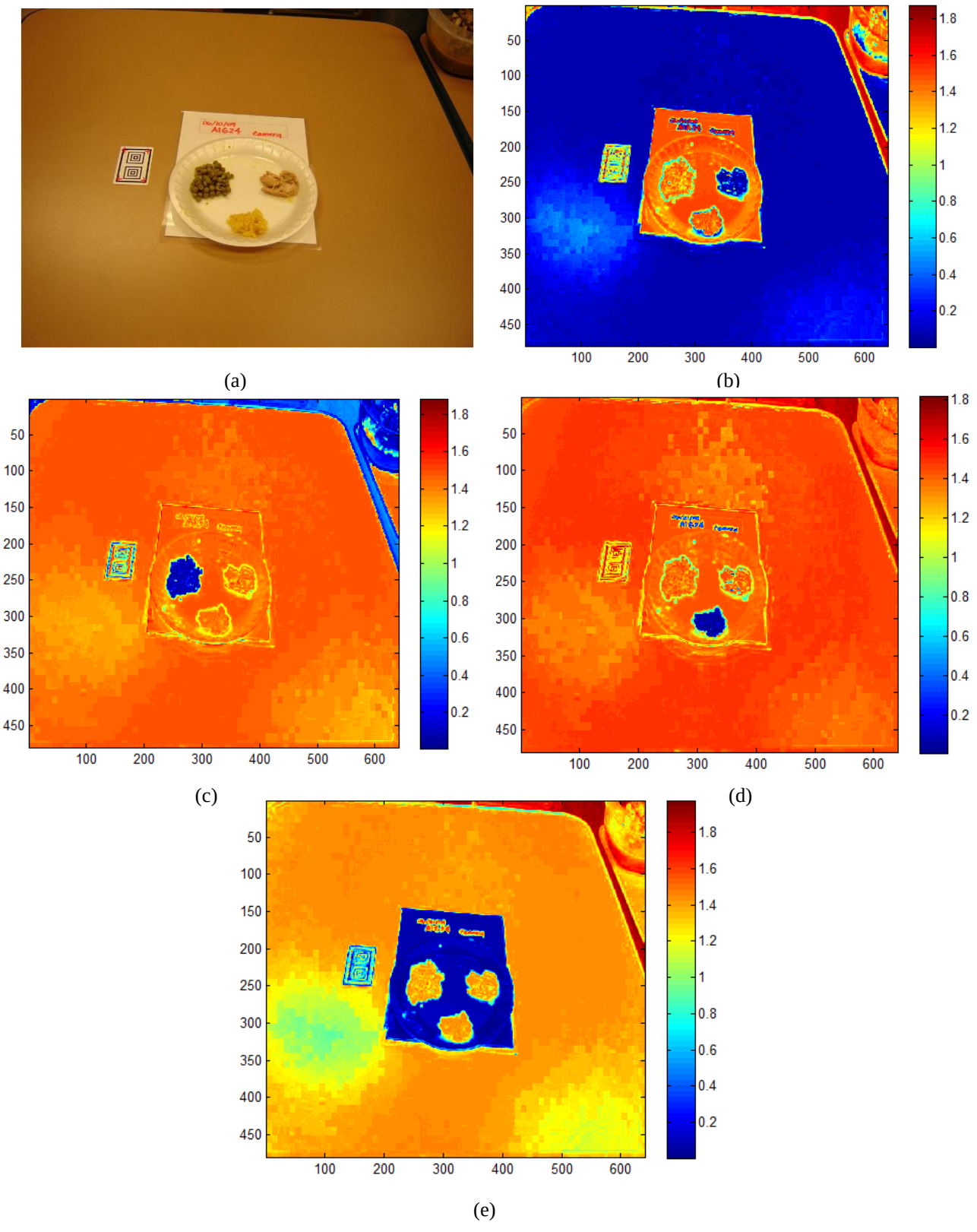
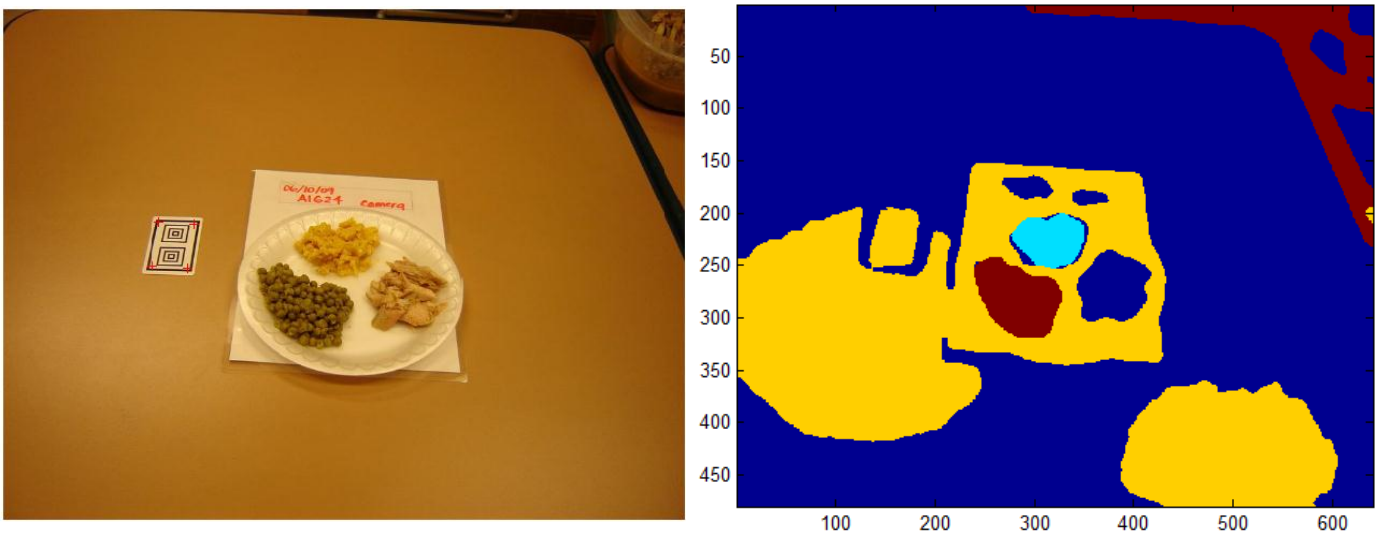
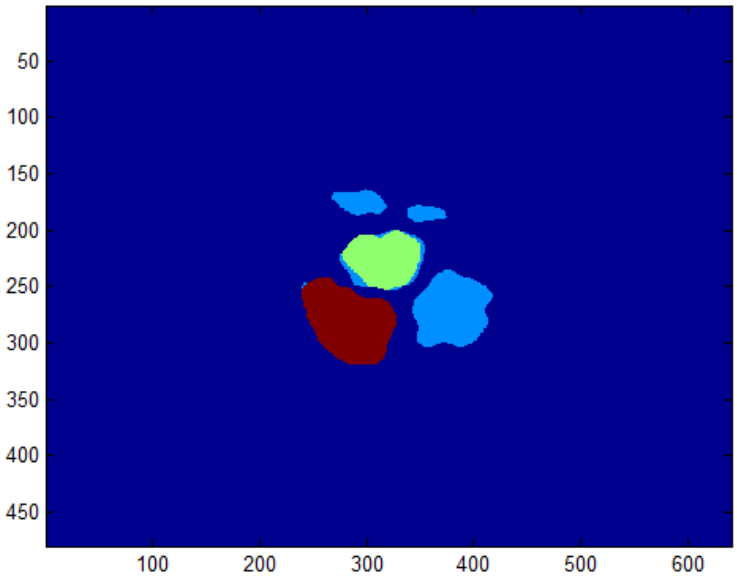


Figure 4.11: Region segmentation and pixel classification by Neural Network classifier (after intake): (a) Input test image (b) Chicken breast pieces (c) Macaroni and cheese (d) Sweet peas (e) Plate.

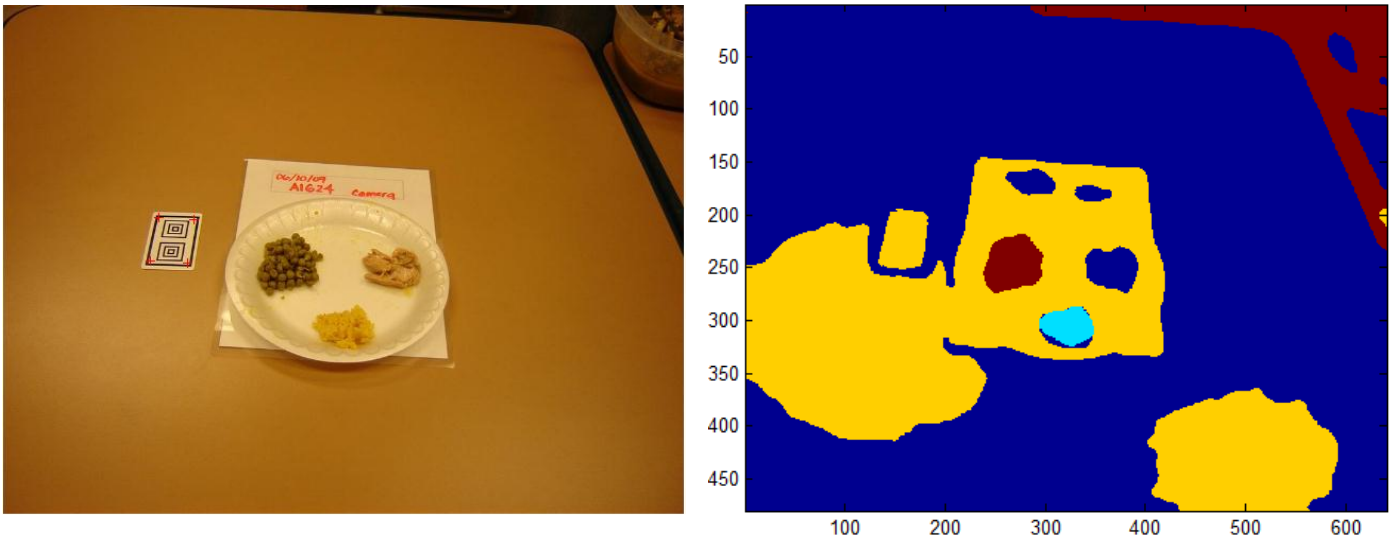


(a) (b)

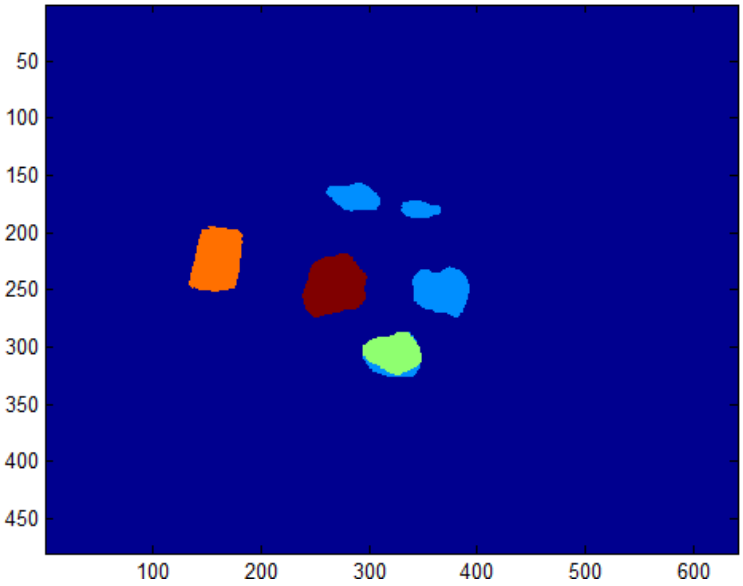


(c)

Figure 4.12: Region segmentation and pixel classification by Mahalanobis distance classifier (before intake): (a) Input test image (b) Segmented and classified image (c) Post processed segmented and classified image.

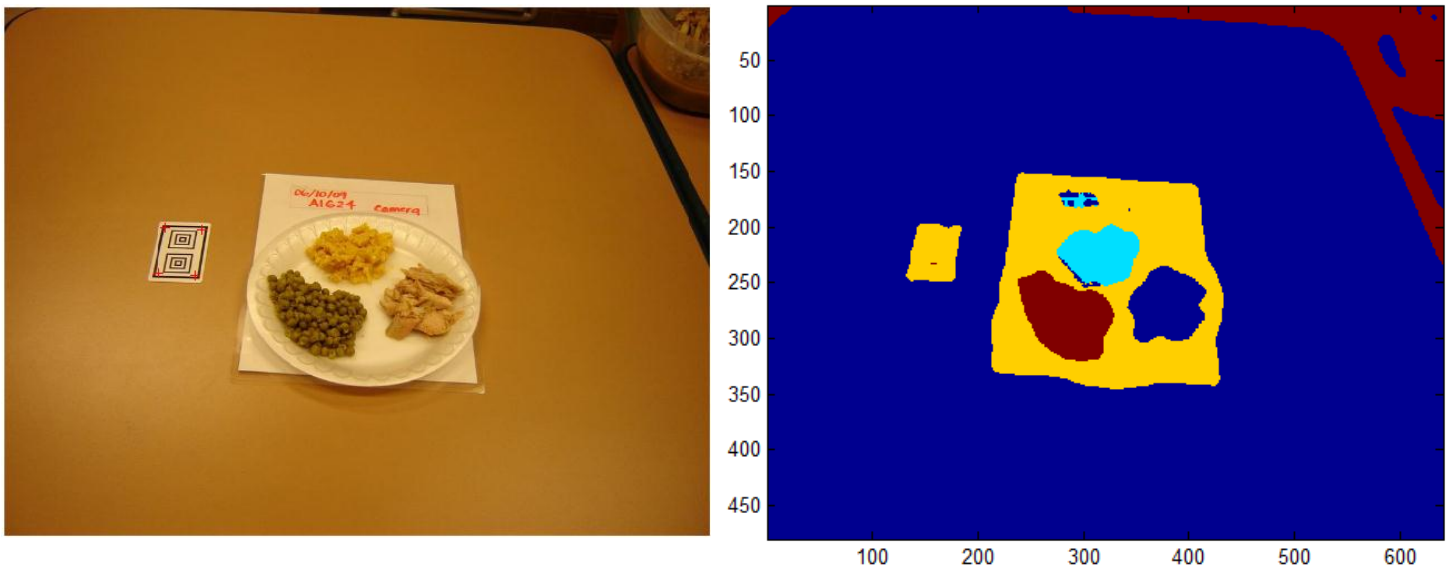


(a) (b)

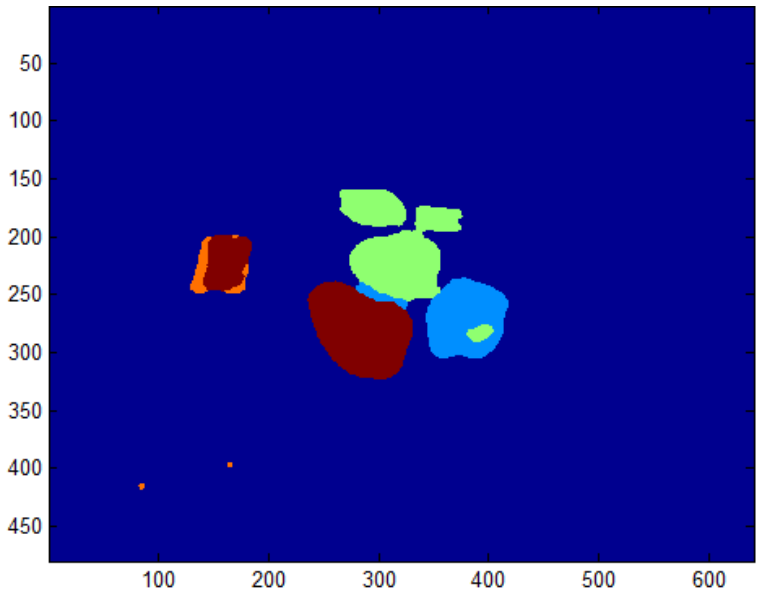


(c)

Figure 4.13: Region segmentation and pixel classification by Mahalanobis distance classifier (after intake): (a) Input test image (b) Segmented and classified image (c) Post processed segmented and classified image.

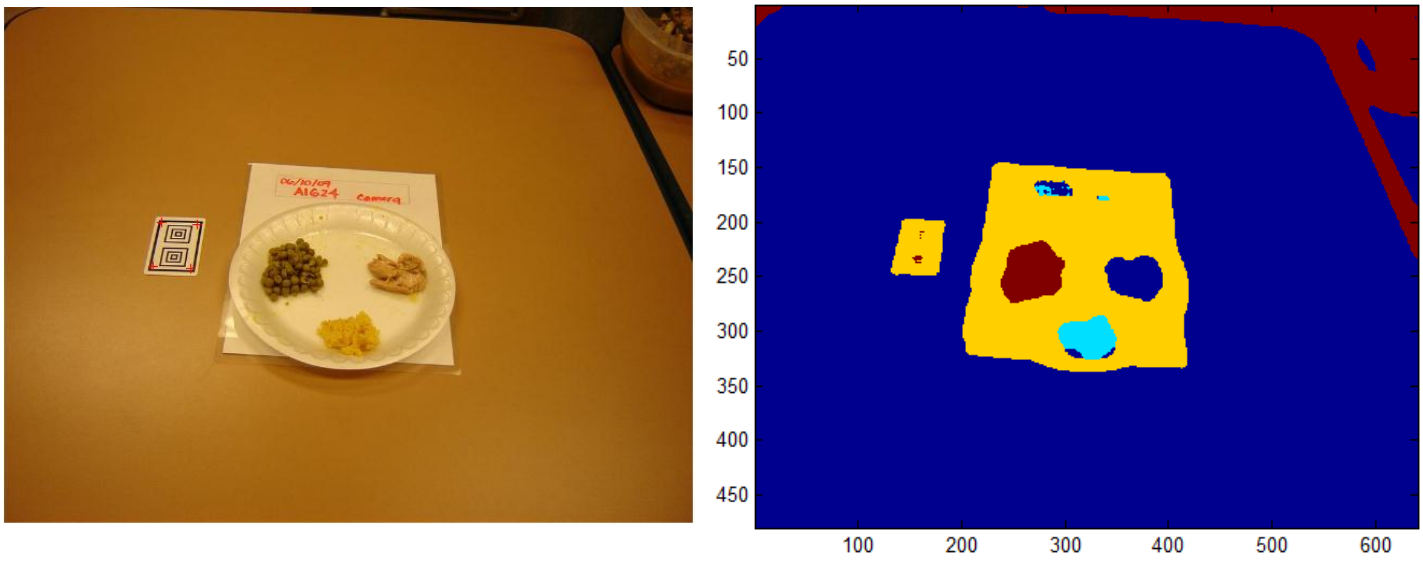


(a) (b)

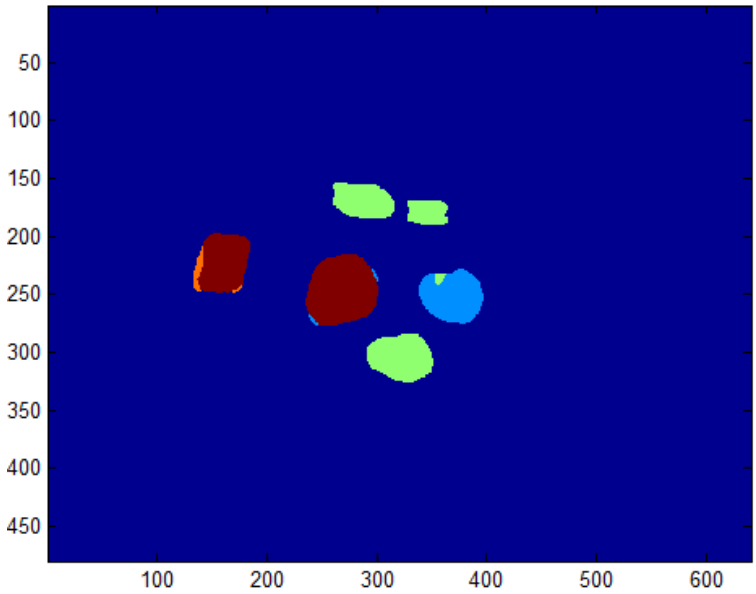


(c)

Figure 4.14: Region segmentation and pixel classification by Neural Network classifier (before intake): (a) Input test image (b) Segmented and classified image (c) Post processed segmented and classified image.



(a) (b)



(c)

Figure 4.15: Region segmentation and pixel classification by Neural Network classifier (after intake): (a) Input test image (b) Segmented and classified image (c) Post processed segmented and classified image.

5. ESTIMATION AND PERFORMANCE ANALYSIS

5.1. Gram Estimation and Performance Analysis

The aim of this section is to address how to compute gram estimation for each food type utilizing segmented and classified image, considering reference gram amount which is entered by user, and integrating reference card corner detection and geometric transformation in the system, as well as describing and analyzing the performance of the existing system. The underlying idea behind of gram estimations is as follows, which is illustrated with details in Figure 5.2. First of all, the reference card corners are applied to geometric transformation to generate the warped card corners. To warp segmented and classified image, the warped card corners, reference card corners, and the segmented and classified image are plugged into geometric transformation function.

In the next step, once we have the number of pixels in both the warped card image and the particular food area, and also real card area (which comes from the Training Stage) in terms of squared inches, then the surface area of the classified particular food is calculated in the unit of squared inches considering correlation between number of pixels and real card area (which is depicted in Figure 5.2). In the last step, with known the ratio of the weight and surface area of the training food, where density is defined as the weight of training food over the surface area of the training food, is multiplied by the surface area of the classified particular food. This operation gives us gram food amount for a particular food class.

As described above, the real area of the food region is determined using the reference card. Based on the association between the food region area and gram amount (equivalently, volume), the amount of food in the picture is estimated. The formula between the area and the volume depends on the food type and shape of the plate used. For some food, the volume is roughly proportional to the area. On the other hand, for some other foods, such as soup, the

shape of the bowl needs to be known to establish the formula between the area and the volume. (See Fig.5.1 for an illustration.) In our current system, we assume linear proportionality between the food area and the volume. As a result, we estimate the area (therefore, the volume and gram amount) of the food in before and after pictures, and estimate the amount of food intake.

The system performance is tested with Baylor Image Sets. The gram amount estimates are shown in Table 5.1. According to the table, the system performed some over estimates for the following food types' after intake photos, garlic toast, spinach, baked beans and rice. On the other hand, the system carried out some under estimates for the following food types' before intake photos, potato salad with mayo, spinach and broccoli with cheese.

We will add a feature to our system to associate the area-volume formula with the food type so that the volumes are estimated more accurately for food that is put in bowls; the system will assume a standard bowl shape to establish the area-volume formula.

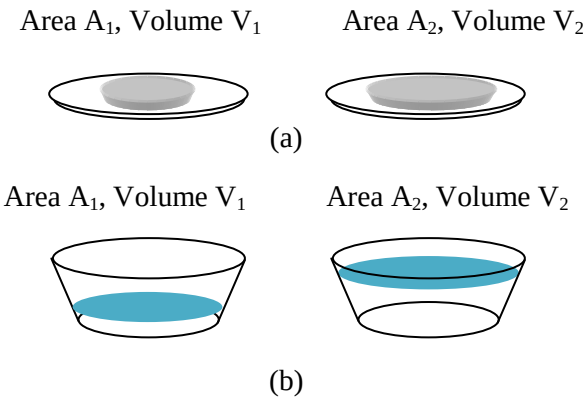


Figure 5.1: Area-Volume relation. (a) Volume is linearly proportional to the area; that is, $V_2 = A_2 (V_1 / A_1)$. (b) The volume depends on the shape (namely, top and bottom areas and the height) of the bowl, which modeled as a cut cone.

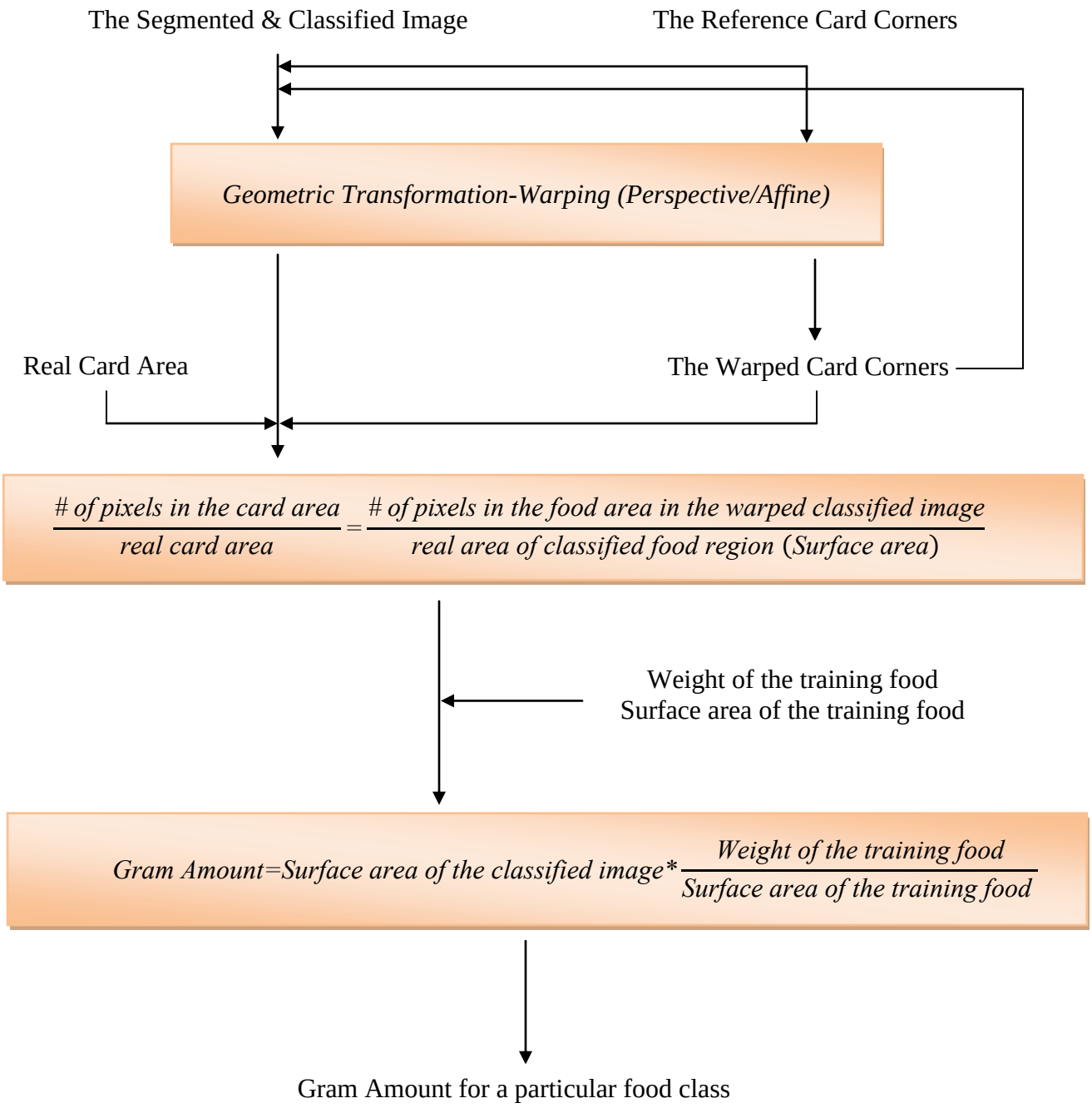


Figure 5.2.: Flowchart of gram amount calculation.

Table 5.1: Comparison of gram estimations and percentage error table

Food Type	Before Intake(g)	After I.(g)	Final I.(g)	Before(correct)	After(correct)	Final I. (correct)	Before Error	After E.	Final E.
Corn on the Cob	136.11861	141.42	0	129.8	84	45.8	5%	0%	0%
Garlic Toast	39.426152	11.995	27.430	38	6.6	31.4	4%	82%	-13%
Lasagna	155.21957	56.770	98.449	142.1	38.4	103.7	9%	48%	-5%
Spinach	49.474755	23.834	25.640	53.6	13.4	40.2	-8%	78%	-36%
Baked Beans	122.31519	48.485	73.830	98.6	15.2	83.4	24%	219%	-11%
Baked Drumstick	121.58870	0	121.58	135.3	25.8	109.5	-10%	0%	11%
Potato Salad w Mayo	72.952025	28.382	44.569	136.3	57	79.3	-46%	-50%	-44%
Enchilada Casserole - whole	171.43673	0	171.43	129.9	0	129.9	32%	42%	32%
Pinto Beans	51.989962	21.261	30.728	39.2	12.5	26.7	33%	70%	15%
Spanish Rice	60.145378	25.741	34.403	45.1	15.8	29.3	33%	63%	17%
Chicken Breast Pieces	56.933028	25.747	31.185	54.5	17.9	36.6	4%	44%	-15%
Macaroni and Cheese	44.230352	24.532	19.697	47.1	16.1	31	-6%	52%	-36%
Sweet Peas	79.279388	38.521	40.757	64.2	27.7	36.5	23%	39%	12%
Broccoli with Cheese	63.006484	20.529	42.476	79.3	17.2	62.1	-21%	19%	-32%
Chili Mac	93.099989	47.820	45.279	88.4	28.5	59.9	5%	68%	-24%
Rice	49.067331	18.952	30.114	32.4	10.4	22	51%	82%	37%
Total							20%	60%	21%

Table 5.2: Weight, Surface Area, and Density for training sets of tested food classes

Chicken Breast Pieces				
Weight	**Surface Area (Affine)**	**Surface Area (Scaling)**	**Weight/Surface Area (Affine)**	**Weight/Surface Area (Scaling)**
106.8	25.74	25.98	4.15	4.11
90.6	24.06	24.49	3.77	3.70
80	23.02	22.68	3.48	3.53
69.6	19.12	19	3.64	3.66
60.7	17.87	17.77	3.40	3.42
38.7	14.8	14.94	2.61	2.59
31.7	9.14	9.24	3.47	3.43
21.3	12.1	12.07	1.76	1.76
11.7	4.51	4.50	2.59	2.60
50.9	14.88	15.38	3.42	3.31
			3.23	3.22
Macaroni and Cheese				
Weight	**Surface Area (Affine)**	**Surface Area (Scaling)**	**Weight/Surface Area (Affine)**	**Weight/Surface Area (Scaling)**
102.5	27.72	27.55	3.70	3.72
91.7	25.99	25.82	3.53	3.55
79.8	23.27	23.61	3.43	3.38
69.4	22.4	22.28	3.10	3.11
59.4	22.1	21.72	2.69	2.74
50.5	18.22	18.12	2.77	2.79
40.6	14.2	14.47	2.86	2.81
30.7	11.82	11.72	2.60	2.62
20.1	6.12	6.22	3.28	3.23
10	3.8	3.78	2.63	2.65
			3.06	3.06
Sweet Peas				
Weight	**Surface Area (Affine)**	**Surface Area (Scaling)**	**Weight/Surface Area (Affine)**	**Weight/Surface Area (Scaling)**
97.6	18.89	18.91	5.17	5.16
88.2	24.41	24.67	3.61	3.58
75.5	15.61	15.76	4.84	4.79
62.9	17.92	18.21	3.51	3.45
50.5	12.76	12.7	3.96	3.98
40.4	14.39	14.45	2.81	2.80
30.2	11.03	10.97	2.74	2.75
20.7	7.77	7.91	2.66	2.62
10.3	4.97	4.95	2.07	2.08
			3.49	3.47

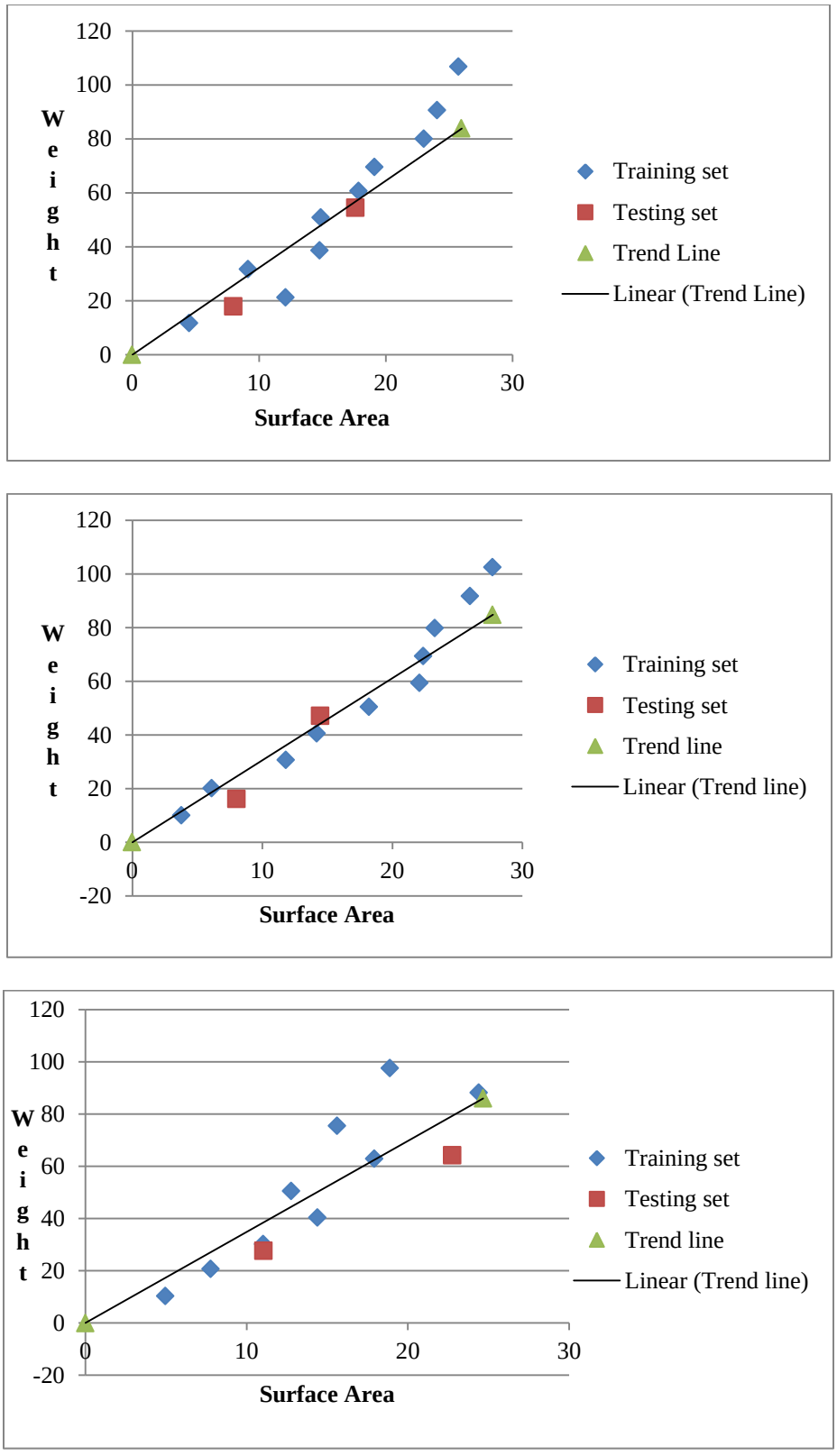


Figure 5.3: Area-surface correlation for various food classes: Chicken breast pieces, Macaroni and cheese, and Sweet peas (from top to bottom).

6. CONCLUSIONS AND FUTURE WORK

Measuring people's energy intake in free living conditions has been challenging for so long. Numerous approaches have been proposed to measure energy intake. The goal of our work is to present a framework to automatically estimate the food intake in free living conditions. We use a reference card system to estimate the true area of the food portions. This gives us the estimated gram amount. Once we have the estimated gram amount, this value is joined with the portion code and portion weight as well as nutritional information in the USDA Food and Nutrient Database for Dietary Studies (FNDDS) to obtain the final energy intake.

The system relies on accurate detection of the corners of the reference. In the event of low-resolution images, the corners may not be estimated accurately; in such a case, we may do a template matching to the reference card or fit lines to the edges of rectangles in the card using Hough transform to eventually have a more accurate estimate of the position of the card. Main contribution of this thesis was to develop an image analysis system to classify and segment each food item. We achieved up to 80% classification accuracy by incorporating optional manual correction into the framework.

Currently, we are using color features, Gabor texture features [37, 38] in our system; these are obviously not sufficient as different foods may have similar color and texture features. We will add various texture features such as DCT, Gray-level co-occurrence matrix [33, 34] to improve the performance of our system. Another possible addition to our system will be use of the Support Vector Machine (SVM) classifier [25]. We are also planning to include image enhancement modules, such as image denoising, compression artifact reduction, and contrast enhancement subsystems, to handle low-quality images. Another future work is to investigate the use of multiple images to construct 3D structures, and to have more accurate estimate of the gram amounts.

REFERENCES

[1] C. L. Ogden, K. M. Flegal, M. D. Carroll, and C. L. Johnson, "Prevalence and trends in overweight among us children and adolescents," *JAMA: The Journal of the American Medical Association,* vol. 288, pp. 1728–32, 1999-2000.

[2] C. L. Ogden, M. D. Carroll, L. R. Curtin, M. A. McDowell, C. J. Tabak, and Flegal K. M., "Prevalence of overweight and obesity in the united states," *JAMA: the journal of the American Medical Association,* vol. 295, pp. 1549–55, 1999-2004.

[3] W.H. Dietz, "Health consequences of obesity in youth: childhood predictors of adult disease," *Pediatrics*, vol. 101, pp. 518-525, 1998.

[4] C. K. Martin, S. Kaya, B. K. Gunturk, "Quantification of food intake using food image analysis," *Engineering in Medicine and Biology Society (EMBC), Annual International Conference of the IEEE,* pp. 6869 – 6872, 2009.

[5] D. A. Williamson, H. R. Allen, P. D. Martin, A. Alfonso, B. Gerald, and A. Hunt, "Digital photography: a new method for estimating food intake in cafeteria settings," *Eat Weight Disord,* vol. 9, pp. 24-8, Mar 2004.

[6] D. A. Williamson, H. R. Allen, P. D. Martin, A. J. Alfonso, B. Gerald, and A. Hunt, "Comparison of digital photography to weighed and visual estimation of portion sizes," *J Am Diet Assoc,* vol. 103, pp. 1139-45, Sep 2003.

[7] P. C. Mahalanobis, "On the Generalised distance in statistics," *Proc of the National Institute of Sciences of India*," vol. 2, pp. 49-55, 1936.

[8] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, Springer, 2006

[9] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Prentice-Hall, 2008.

[10] M. B. Livingstone and A. E. Black, "Markers of the validity of reported energy intake," *J Nutr,* vol. 133 Suppl 3, pp. 895S-920, March 2003.

[11] D. A. Schoeller, "How accurate is self-reported dietary energy intake?" *Nutr Rev,* vol. 48, pp. 373-9, Oct 1990.

[12] L. de Jonge, J. P. DeLany, T. Nguyen, J. Howard, E. C. Hadley, L. M. Redman, and E. Ravussin, "Validation study of energy expenditure and intake during calorie restriction using doubly labeled water and changes in body composition," *Am J Clin Nutr,* vol. 85, pp. 73-9, Jan 2007.

[13] A. H. Goris, M. S. Westerterp-Plantenga, and K. R. Westerterp, "Undereating and underrecording of habitual food intake in obese men: selective underreporting of fat intake," *Am J Clin Nutr,* vol. 71, pp. 130-4, Jan 2000.

[14] L. G. Bandini, D. A. Schoeller, H. N. Cyr, and W. H. Dietz, "Validity of reported energy intake in obese and nonobese adolescents," *Am J Clin Nutr*, vol. 52, pp. 421-5, Sep 1990.

[15] D. A. Schoeller, L. G. Bandini, and W. H. Dietz, "Inaccuracies in selfreported intake identified by comparison with the doubly labeled water method," *Can J Physiol Pharmacol*, vol. 68, pp. 941-9, Jul 1990.

[16] J. Beasley, W. T. Riley, and J. Jean-Mary, "Accuracy of a PDA-based dietary assessment program," *Nutrition*, vol. 21, pp. 672-7, Jun 2005.

[17] M. Nelson, M. Atkinson, S. Darbyshire, "Food photography II: use of food photographs for estimating portion size and the nutrient content of meals," *Br J Nutr*, vol. 76, pp.31-49, 1996.

[18] D. A. Williamson, H. R. Allen, P. D. Martin, A. Alfonso, B. Gerald, and A. Hunt, "Digital photography: a new method for estimating food intake in cafeteria settings," *Eat Weight Disord*, vol. 9, pp. 24-8, Mar 2004.

[19] D. A. Williamson, H. R. Allen, P. D. Martin, A. J. Alfonso, B. Gerald, and A. Hunt, "Comparison of digital photography to weighed and visual estimation of portion sizes," *J Am Diet Assoc*, vol. 103, pp. 1139-45, Sep 2003.

[20] USDA, "United States Department of Agriculture, Agricultural Research Service, Continuing Survey of Food Intakes by Individuals," 1994-1996, 1998, 2000.

[21] D. H. Wang, M. Kogashiwa, S. Ohta, S. Kira, "Validity and reliability of a dietary assessment method: the application of a digital camera with a mobile phone card attachment," *J Nutr Sci Vitaminol (Tokyo,)*;48:498–504, 2002.

[22] F. Zhu, A. Mariappan, D. Kerr, C. Boushey, K. Lutes, D. Ebert, and E. Delp, "Technology-assisted dietary assessment," *Proceedings of the IS&T/SPIE Conference on Computational Imaging VI*, vol. 6814, San Jose, CA, January 2008

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

[24] R. Duta, P. Hart, and D. Stork, *Pattern Classification*, New York, NY: John Wiley and Sons INC, 2001.

[25] N. Cristianini and J. Taylor, *An introduction to support vector machines*, Cambridge: Cambridge University Press, 2000.

[26] C. J. Boushey, D. A. Kerr, J. Wright, K. D. Lutes, D. S. Ebert, E. J. Delp, "Use of technology in children's dietary assessment," *Eur J Clin Nutr*, vol. 63, pp. 50-57, 2009.

[27] A. Mariappan, M. B. Ruiz, F. Zhu, C. J. Boushey, D. A. Kerr, D. S. Ebert, E. J. Delp, "Personal dietary assessment using mobile devices," *Proc SPIE Intl Soc Optic Eng*,vol. 7246:72460Z, pp. 1-12, 2009.

[28] C. K. Martin, H. Han, S. M. Coulon, H. R. Allen, C. M. Champagne, and S. D. Anton, "A novel method to remotely measure food intake of free-living people in real-time: The Remote Food Photography Method (RFPM)," *Br J Nutr,* vol. 101, pp. 446-456, 2009.

[29] A. A. Stone, and S. Shiffman, "Ecological momentary assessment (EMA) in behavioral medicine," *Ann Behav Med*, vol. 16, 1994.

[30] I. Woo, K. Otsmo, S. Y. Kim, D. S. Ebert, E. J. Delp III, C. J. Boushey, "Automatic portion estimation and visual refinement in mobile dietary assessment," *Electrical Imaging science and technology, Computational Image VIII*, San Jose, CA, January 2010.

[31] D. G. Chandler, E. P. Batterman, and G. Shah, "Hexagonal, Information Encoding Article, Process and System," *US Patent No.4874936*.

[32] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," *Proc. of the 4th Alvey Vision Conf.,* pp. 147-151, 1988.

[33] B. Jahne, *Digital Image Processing*, Springer, 2008.

[34] http://www.wikipedia.org

[35] T. M. Mitchell, *Machine Learning*, McGraw-Hill International Editions, 1997.

[36] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, Boston: PWS Publishing Co.,1996.

[37] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using gabor filters*," in Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, (Los Angeles, CA), November 1990.

[38] J. R. Movellan, "Tutorial on Gabor Filters", September 3, 2002.

VITA

Sertan Kaya was born in Kayseri, Turkey, the son of Yasar and Nurhayat Kaya. After completing the high school in Nigde, he attended the Department of Electronics Engineering at Gebze Institute of Technology (GIT), Gebze, Kocaeli, Turkey, where he received his Bachelor of Science degree in 2005. He was admitted to the Department of Electrical and Computer Engineering at Louisiana State University, Baton Rouge, Louisiana, in August 2008. He joined the image processing lab under the supervision of Dr. Bahadir K. Gunturk. His research interests are computer vision, pattern classification and object recognition. During his assistantship period, Dr. Gunturk's group collaborated with the Bioengineering Group at Pennington Biomedical Research Center, Baton Rouge, Louisiana, and Sertan published one conference paper in the Institute of Electrical and Electronics Engineers (IEEE) with Dr. Gunturk.