RC 20470 (05/29/96)
Computer Science

IBM Research Report

VeggieVision: A Produce Recognition System

R. M. Bolle, J. Connell, N. Haas, R. Mohan, G. Taubin

IBM Research Division
T.J. Watson Research Center
Yorktown Heights, New York

Research Division
Almaden · T.J. Watson · Tokyo · Zurich

VeggieVision: A Produce Recognition System[1]

Ruud M. Bolle, Jonathan H. Connell, Norman Haas,
Rakesh Mohan, and Gabriel Taubin

Exploratory Computer Vision Group
IBM T. J. Watson Research Center
P.O. Box 704
Yorktown Heights, N.Y. 10598
(Address e-mail to: bolle @ watson.ibm.com)

May 1996

Abstract

Checking out produce poses an enormous problem for grocery stores and supermarkets. Essentially, the technology of produce checkout has not changed in the last 1,000 years: the checker visually identifies the produce and weighs it to determine the price.

It is a well-established fact that the produce department is a significant factor in how customers rate the quality of a store. Produce is also associated with many problems, including: 1) shrinkage [losses], 2) slow checkout, 3) checker training, 4) affixing price look-up code labels, 5) expensive packaging, 6) solid waste. Because of these problems, most grocers and supermarket chains will admit in private that they would rather not carry produce.

We present an automatic produce ID system ("VeggieVision"), intended to ease the produce checkout process. The system consists of an integrated scale and imaging system with a user-friendly interface. When a produce item is placed on the scale, an image is taken. A variety of features, color, texture (shape, density), are then extracted. These features are compared to stored "signatures" which were obtained by prior system training (either on-line or off-line). Depending on the certainty of the classification, the final decision is made either by the system or by a human from a number of choices selected by the system. Over 95% of the time, the correct produce classification is in the top four choices.

Object Recognition, Segmentation, Color, Learning in Computer Vision, Systems and Applications

[1]This paper has been submitted to WACV 1996. *This material has never been published and there is no concurrent submission of this work.*

1 Introduction

"Consumers still realize that produce departments feature the healthiest items in the supermarket," says George Dunlop, president of the United Fresh Fruit and Vegetable Association (UFFVA) [1] "They're now more aware of the fact that the number one health problem in this country is obesity, and that eating more fruits and vegetables on a daily basis can significantly lessen that problem." Hence the produce department is an important section of the supermarket. The quality and the quantity of items in the produce department also is a significant factor in how customers regard a store and whether they will bring in repeat business. Store owners and customers alike would like to see an increased variety of produce items. However, there is a number of problems associated with carrying produce, and produce checkout is one of the major ones.

In the last 25 years, packaged goods checkout has been greatly expedited by bar code readers for UPC (Universal Product Code) labeled items. Produce checkout, on the other hand, has not changed much since the first appearance of grocery stores. At the point of sale (POS), the cashier has to recognize the produce item(s), weigh or count them, and determine their price. For many items sold in bulk, this is done by manually entering the non-mnemonic PLU (Price Look Up) codes associated with the items. These may be available at the POS in the form of a printed list or a booklet with pictures. Some produce or bulk items are identified by PLU labels affixed to them, but attaching the labels is labor-intensive, and customers have to peel off the labels before consumption. Pre-packaged produce items may be labeled, but packaging, in general, is not considered a good solution: it increases cost, creates solid waste, and the customer is unable to inspect the product.

Produce checkout has multiple problems:

Losses incurred by the store – shrinkage: *Mis-rings*, i.e., cashiers unintentionally entering the wrong code number, work against the store, because the customer may point them out only if they are not in the customer's favor. Intentional mis-rings for friends and relatives (*sweethearting*) also result in losses.

Reduced throughput (checker productivity): Checkout tends to be slower for produce than for bar coded goods. Often the checkout person cannot identify a particular item and has to ask what the code, price, or the name of the produce is.

Checker training and turnover: For rapid produce checkout, the cashiers should know the produce codes by heart. Hence, every new checker has to learn, first, to recognize the produce carried by the store and what it looks like, and then, memorize the produce code list. Frequent turnover of employees enlarges this problem.

Customer self-checkout: There is a trend toward having customers do their own checkout. An important component of this process will be a mechanism for the automatic identification or verification of the product.

In this paper, we present a trainable produce recognition system for supermarkets and grocery stores, which addresses these difficulties. The system, which is inexpensive and fast, is a vision system with a single color camera that is built on top of a scale that weighs the produce. From the produce image, multiple recognition clues: color, texture, size, shape, and density (weight/area) are extracted, and integrated to classify the produce. If its identity cannot be uniquely determined by the system, the produce recognition system displays the top recognition choices, from which the human operator makes the final decision. The prototype recognition system is designed with standard off-the-shelf hardware that can be added to existing (PC-based) cash registers. Next generation systems will operate with a card camera and a custom designed image processing board.

The system has been extensively tested on produce from several supermarkets. It achieves a classification success rate of about 84% for the top choice and about 95% for the top four choices. Intra-store and inter-store testing was performed, and experiments on alternative strategies for prototype learning were done.

2 System considerations

Supermarkets and grocery stores are hostile imaging environments and robust, rugged systems are needed. Further, there are system requirements regarding space, speed, price and performance. The following issues are important:

Segmentation: The produce item will have to be imaged against some background, which will vary over time and between checkout stations both within and across stores. Therefore, reliable foreground/background segmentation is required. The use of plastic bags to package produce makes this more difficult.

Color constancy: The color of an object in an image critically depends on the color of the illuminating light, which is highly variable in stores. A controlled illumination system therefore seems unavoidable for achieving color constancy.

Specular reflection: Although specular reflection may provide useful shape information, it does not reflect the natural color of the produce. It should therefore be filtered out – especially when the specular reflection is due to the bag.

Recognition speed: Recognition should be achieved in times comparable to those of current bar code reading. That is, about 1 second per item (including image acquisition) should be achieved. Because supermarket operate on small profit margins, the hardware must be cheap.

Recognition performance: System accuracy should be at least as good as that of the average checker. Performance comparable to bar code scanning (very nearly 100 %) is desirable (but probably not achievable). Therefore, as many cues as possible must be used, to achieve the best possible recognition performance.

Ease of use: The produce ID system should be simple and intuitive to operate, requiring minimal operator training.

System training: In the store, the system should be able to adapt automatically to changes in produce appearance (due to season, freshness, supplier, etc) through incremental learning.

Supermarket environment and ergonomics: To be space-efficient and ergonomically correct, whether installed at the store front end or in the produce section, the produce ID system should be integrated with the bar code reader in a single enclosure. This also allows retrofitting into existing checkout stations.

Database size: Figure 1 shows the average number of produce items that was carried in 1992 by U.S. grocery stores.[2] The numbers vary by store, region, and season; the highest is on the order of 350 items. Of course, no two produce items of the same kind look exactly alike even if picked on the same farm at the same time; appearance

[2]Source: Produce Marketing Association.

		Average	Range
Mid-Winter	Fruits	53	15–150
	Vegetables	85	30–155
Mid-Summer	Fruits	62	6–180
	Vegetables	85	35–170

Figure 1: Number of produce items available in stores

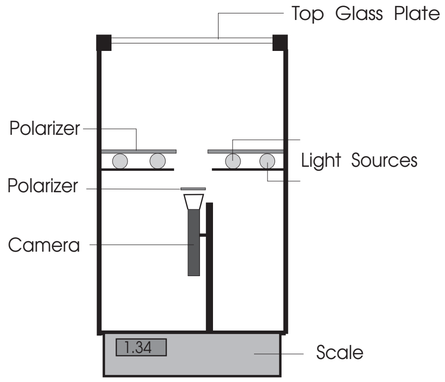
variations are exacerbated by variations in region, season, wholesaler, shipping and storage duration and condition. Rather than attempt to deal with some of these variations by modeling, e.g., ripening process, we have chosen to design a system that is easily trained on the produce inventory of a store and then incrementally adapts to changes in produce appearance. It also permits easy inclusion/exclusion of seasonal items.

3 Imaging and Segmentation

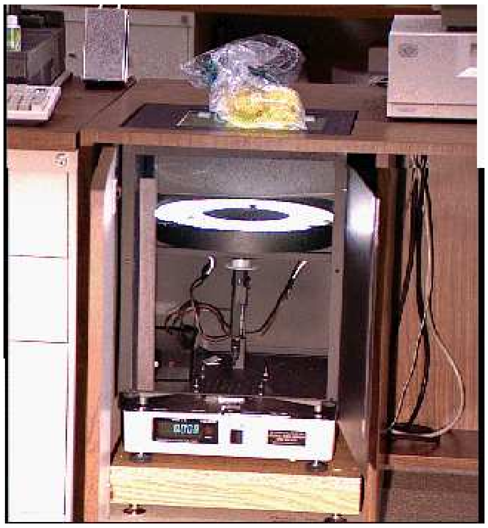
3.1 Imaging Setup

To respond to the challenges described in the previous section, we have designed an upward-looking imaging system, with an opaque enclosure with a transparent top surface (a glass window, approx. 8" $\times$ 9"). The produce rests on the glass window while being imaged. This setup can be easily integrated with a scale, for concurrent weighing, and could in principle also be integrated with a bar code reader, to make a very compact, highly functional package.

Figure 2a shows a cross section of the prototype image acquisition system. In this



(a)



(b)

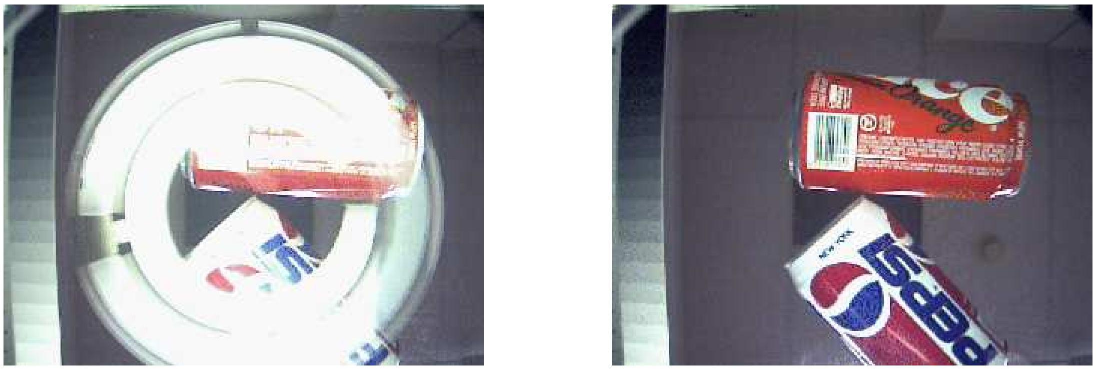
Figure 2: The prototype imaging setup.

setup, two circular fluorescent bulbs are used to illuminate the produce items as uniformly as possible. A photograph of the system is shown in Figure 2b.

The system includes a linear polarizing filter covering the internal light sources, and a second polarizing filter on the camera, orthogonal to the first[2]. It is well known[3] that if light is normally incident on a surface, any specular reflection of this light will preserve its plane of polarization, while any diffuse reflection of it will be unpolarized.[3] So, in our system, the part of the produce which is visible to the camera is illuminated almost entirely by polarized light which will be filtered out if specularly reflected, but not if reflected diffusely. The system obtains good color images of shiny objects, without glare.

Figure 3 shows the effect of this filtering. Figure 3a was taken with the polarizing filters parallel, while Figure 3b was taken with the filters perpendicular. In the latter

[3]If the object is a non-metallic, non-crystalline dielectric, which is the case for produce.



(a) (b)

Figure 3: Parallel polarization (a), perpendicular polarization (b).

configuration, they largely eliminate, not only the specular reflections from the objects, but also the reflection of the light source in the glass plate.

3.2 Segmentation

As mentioned, the imaging system contains a light source. The illumination of each object visible to the camera is the sum of its ambient illumination and that illumination of the system light source. The latter is inversely proportional to the square of the distance to the light. The fraction of incident light reflected to the camera is also inversely proportional to the square of (approximately) this same distance, because of the geometry of the system.

To segment the produce image from the background, the system first takes a picture with the lights off, and then one with the lights on. (See Figure 4.) Pixels which have increased in brightness by more than some threshold T_Δ are tentatively classified as foreground (i.e., produce), the rest as background. The system then examines all the tentative foreground pixels; if, in the original "dark" image, they were brighter than another threshold, T_{dark}, they are re-classified as background. The remainder constitute the "true" foreground. Background pixels are set to zero.

The T_{Δ} test effectively segments out distant objects, such as the ceiling. But it is not enough, because if the produce should happen to be contained in a plastic bag, the bag would be segmented in as foreground. So the T_{dark} test is necessary, and identifies the bags, which are illuminated somewhat by light transmitted through them.

Highly reflective objects will show a greater increase in brightness than less reflective ones equally far away, so in theory, the former might be perceived as closer. But in practice, the brightness variation due to reflectance is much less than that due to distance. Suitable thresholds can be chosen, given knowledge of the ambient conditions, so that even dark vegetables like eggplant show enough brightness variation (provided that it is enclosed by a somewhat milky bag) to classify as foreground. The problem is the dynamic range of today's affordable cameras.

For good segmentation, it is important that the produce be stationary during imaging. This is easily assured because the scale will not give a reading until its platform has stabilized.

The segmented image obtained from the images in Figure 4 is shown in Figure 5. Note that segmented out of this image are: the bag that surrounds the produce, the recessed fluorescent ceiling lights, and the otherwise dark ceiling. A large variety of such ambient light source configurations can be tolerated – ceiling spotlights shining straight into the camera can cause problems, however.

(a) (b)

Figure 4: A dark image (a) and a light image (b) of a produce item.

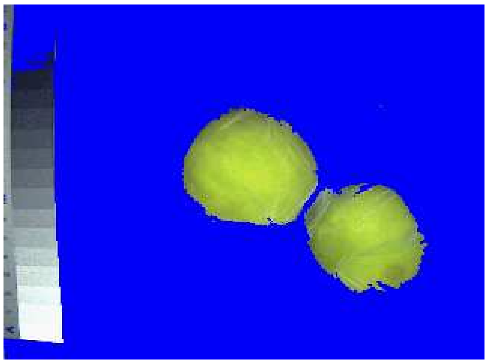


Figure 5: The segmentation of image in Figure 4

3.3 User Interface

Figure 6 shows a graphical display of the user interface of the recognition system. This is a touch sensitive screen on which small pictures of the highest-ranked produce choices are displayed, so that the operator can do visual matching. How many choices the system displays and when it displays these choices or just classifies the produce as the highest ranking choice is customizable.

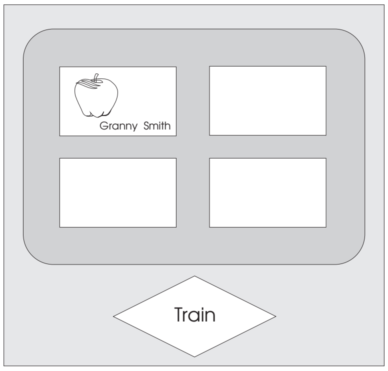


Figure 6: User interface to select right choice.

The human operator can be always in the loop, or it can be set so that choices are only displayed when the selections are unreliable. The number and order of the displayed choices are system variables; possible selections are: most likely $\cdots$ least likely; least expensive $\cdots$ most expensive, etc. The `Train` button allows the operator to request the system to train on a specific produce instance, but, as described in Section 5, if the training example is too close to an existing prototype, it will not be used. If it is used, it will replace the "least useful" prototype of the class (see Section 5).

4 Features for Recognition

For the design of the produce classification system, the issue is not which features to use – color, texture, shape, etc. are the obvious choices – but how to tailor the features and their representations to suit the application.

Note that the representation for a produce item should be invariant with respect to rotation and translation, and with respect to the number of produce items presented, but not to their size. (Oranges should be identified the same, regardless of placement or number, but large oranges are sometimes priced differently from small ones. If a produce count is needed, it is best to do this count after produce classification rather than incorporating the count in the produce representation.) Secondly, because it will be necessary to (re-)train at the point of sale, the representations and the classification mechanism should be simple.

4.1 Histograms

For these reasons, extensive use is made of histograms as produce representation. A histogram is a very compact representation of an object, many orders of magnitude smaller than the raw image. Color histogramming as an identification technology is a practice of long standing; a recent example is [4]. The histogram representation can be extended to other visual cues, as well. To achieve invariance with respect to the number of produce items, our histograms, unlike those in [4], are *normalized* with respect to the foreground (produce portion) of the segmented images.

If histogramming is to be employed, certain conditions must be true, which the design of our system assures or strongly promotes:

- All the training histograms and the recognition histogram should be obtained from images acquired under similar illumination conditions.
- Most of the object is in the image, and is not obscured by other objects.
- It is necessary to know which pixels constitute the object, or at least there should be no distractions in the background (i.e., the image is segmented).

4.2 Color

Color captures a salient aspect of the appearance of produce, and is not dependent on the position or orientation of the produce. Many color descriptions (spaces) can be found in the literature, including:

- the Red/Green/Blue (RGB) space[5],
- the opponent color space [6],
- the Munsell (HVC) space [7],
- the Hue/Saturation/Intensity (HSI)[5] space, similar to Munsell.

Our system builds its color histograms in the three-dimensional HSI space. Hue is the spectral shade, which varies continuously from red through green to blue, saturation is the "depth" or "strength" of the color, and intensity is the brightness or gray level. We convert the camera output to HSI, using the standard transform as can be found in [5]. The histograms for each of H, S and I dimensions are computed as separate, one-dimensional histograms. Then they are concatenated into one long, one-dimensional, "extended" histogram.

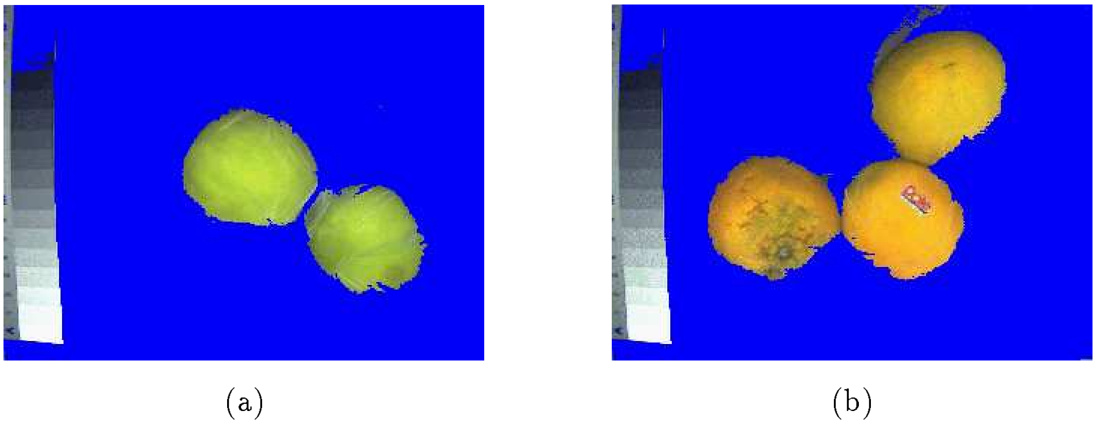


Figure 7: Apples (a) and oranges (b).

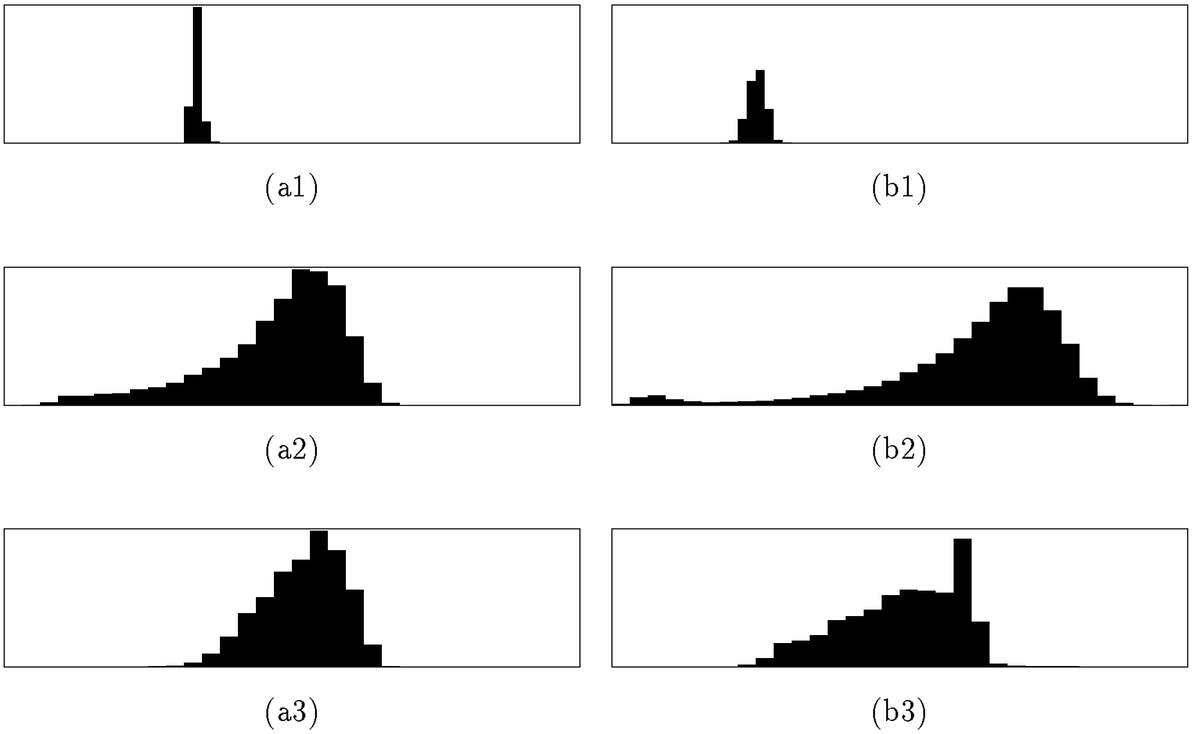


Figure 8: Histograms: hue (1), saturation (2), intensity (3).

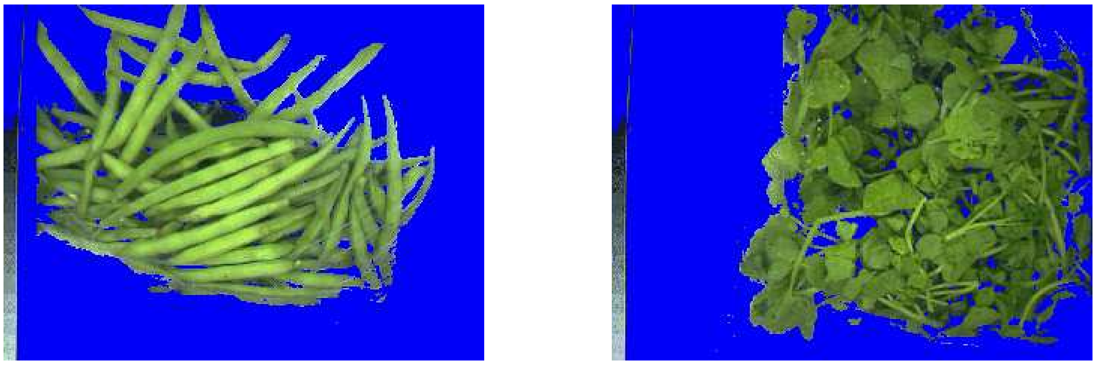
Figure 7 shows two examples of segmented images: Granny Smith apples (Fig. 7a) and oranges (Fig. 7b). Figure 8 shows the corresponding histograms, produced by accumulating a count of the quantized values of the H, S and I components of each pixel in the segmented image's foreground. Note that the most profound difference between the apples and the oranges is in the hue component. The peak for apples is to the right of the peak for oranges, reflecting the fact that apple hue lies in the green part of the spectrum. The saturation histograms show that the oranges' color is a little stronger than that of the apples. In Section 5, we will discuss how much the different components of color contribute to overall recognition performance.

4.3 Texture

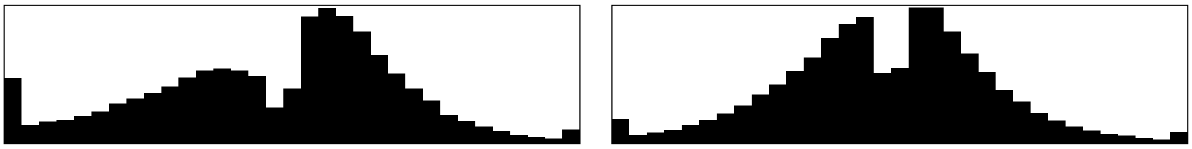
Texture is important for discriminating produce, for there are a great many green vegetables which are not reliably discriminable by color.

Texture is a visual feature that is much more difficult to describe and to capture computationally than color; also, it is a feature that cannot be attributed to a single pixel, but rather to a patch of an image. It is a description of the spatial brightness variation in that patch. Texture can be a repetitive pattern of a common unit (a *texel*), as on artichokes and pineapples, or it can be more random, as with the leaves of parsley – compare to the *structural* and *statistical* texture descriptions in [8]. Much research has been done on texture in the past thirty years, and many computational texture measures have been developed [8, 9].

We have experimented with many textures measures that have been proposed, ranging from Tamura [10] to Laws [11]. Though they may perform quite well on the Brodatz texture database, sadly, none of them performs well for distinguishing produce, and they require far more computation than this application can afford. We have developed two texture measures (measures A and B), each of which can be computed extremely quickly and outperforms all other measures we have experimented with in discriminating between produce textures. Each texture measure is computed using the segmented image from the green channel output only.



$(a1)$ $(b1)$



$(a2)$ $(b2)$

Figure 9: Texture Measure A histograms on (a) String Beans and (b) Watercress: (1) Segmented Image, (2) Magnitude.

Texture measure A: This measure convolves the image with two crossed bar masks, with the bars parallel to the image x and y directions. The arms are of equal lengths, one pixel wide. The horizontal and vertical convolution of the image with these bars is denoted by $C_h(x, y)$ and $C_v(x, y)$, respectively, with pixel (x, y) the center point.

From this, a texture magnitude $M(x, y)$

$$M(x,y) \quad = \quad \sqrt{C_h(x,y)^2 + C_v(x,y)^2} \tag{1}$$

is computed.

The bar masks are of the form $[-1 \quad 2 \quad -1]$, $[-1 \quad -1 \; 2 \; 2 \quad -1 \quad -1]$, etc. The convolutions are performed for a few different sizes of mask, against the full-resolution image. A histogram of the magnitude values for all pixels is computed and concatenated with the color histograms.

Figure 9 shows segmented images of string beans and watercress and gives their Measure A histograms.

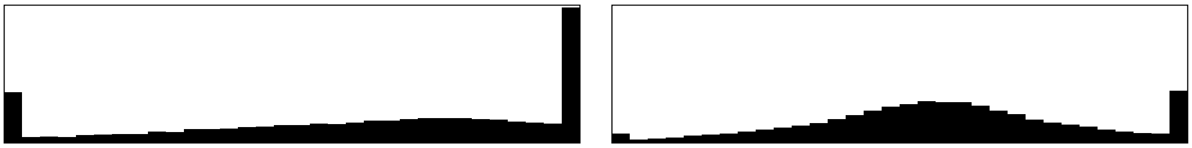
Texture measure B: Measure B is a "center-surround" operator, a kind of first-order statistic. It consists of computing and histogramming the deviation of the image intensity of a pixel from the average of its neighbors in a moderate-sized block centered on that pixel. (For reasons of speed, it is performed on a reduced image, obtained by subsampling the full-resolution one.) The deviations are histogrammed.

As with the color histograms, this histogram is normalized for foreground pixel count, i.e., object size. Since this measure is roughly radially symmetric, it is approximately rotation-invariant.

Figure 10 gives the Measure B histograms for the same items as Figure 9.

4.4 Other Clues

The system can also use other clues: shape, size and density, which are not discussed here for lack of space. Size and shape are visual features that are extracted from the produce



(a) (b)

Figure 10: Texture measure B on string beans (a) and watercress (b).

boundary. Density, extracted from produce weight, on the other hand, is a disparate source of information; using density as an additional clue for recognition would be sensor fusion, in the true sense of the word. See [12] for more details.

5 Classification and Training

Recognition, using the prototypes in the database, should be fast. The prototypes should be easy to acquire, and to re-aquire, if the items to be recognized change in appearance.

The system uses nearest-neighbor techniques (e.g., [13]) for classification. It thus avoids complicated dynamic updating of its database and classification rules that would be required for possibly more efficient and more sophisticated data classification schemes, yet it is still fast enough that the DSP hardware can scan $\approx O(4,000)$ prototypes [$O(400)$ classes $\times$ $O(10)$ prototypes per class] in under 1 second (400 produce items is the high-end of what is found in supermarkets; see Figure 1). Details of the classification process follow.

Let P^i, $i = 1, \cdots, N$ denote the N prototype histograms, and let Q denote the histogram of a produce item to be recognized. Each (extended) histogram is a concatenation of its component feature histograms. P^i_f and Q_f refer to these components; $f \in F = \{$h(hue), s($saturation$), i($intensity$), t($texture$)$\}$. (Each is normalized as appropriate for that feature.) With each prototype P^i, a produce class identifier $I(P^i)$ (for instance, "Arugula") is associated.

Comparison between Q and the P's is performed using a distance measure. The distance between two histograms is the weighted sum of the distances between the histogram components, where the component feature histograms have weights w_f:

$$d^i \equiv d(Q, P^i) \;=\; \sum_{f \in F} w_f \, d(Q_f, P^i_f), \tag{2}$$

and the component distances are L_1 ("Manhattan") distances between the histogram vectors. This provides a straightforward method of integrating the various features, it is computationally simple, and has been validated by experimentation.

Q is classified as:

$$< identity > \equiv \; I(P^i) \mid d^i, \; i = 1, \cdots, N, \text{ is minimum.} \tag{3}$$

The recognizer reports a decision qualifier in the form of "sure," "okay," or "unreliable," and

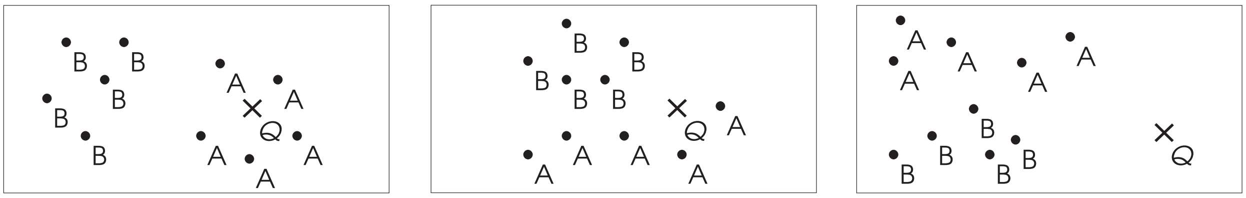
- the unique identification, if the qualifier is "sure," or
- multiple choices, if the top choice is "okay" or "unreliable."

This incorporates distances (uncertainties) and the prototype distribution in feature space into the classification, and takes advantage of the fact that a human operator is (or can be asked to be) in the decision loop. The three situations described below are depicted in Figure 11. With respect to a distance threshold T and a count n:

a: If $d^j < T$, $j = 1 \cdots n$, and $I(P^j) = I(P^1)$, $j = 2 \cdots n$, then Q lies in a portion of feature space populated only by prototypes of a single class, and the classification is judged to be "sure."

b: If $d^j < T$, $j = 1 \cdots n$, and $I(P^j) \neq I(P^1)$, for some $j = 2 \cdots n$, the classification is labeled "okay."

c: If $d^1 > T$, the classification is "uncertain," because Q is too dissimilar from the nearest prototype.



a: "sure" b: "okay" c: "uncertain"

Figure 11: Classification qualifiers based on feature space.

"Sure" classification sales could be rung up without human operator intervention. For "uncertain" and "okay" classifications, the class identities $I(P^j)$, $j \in 1, \cdots, N$, choices are computed and displayed (in order of closest match) to the operator, who can endorse the top classification, select a choice from multiple choice menu, or override the classification.

If the operator indicates that the classification made by the system was wrong, the system retrains, using Q as a new prototype of the class indicated by the operator. If, instead, the classification was correct but the system was not sure, it also incrementally retrains, using Q as a prototype of the class it selected. Finally, if the matcher generates the correct answer and is sure of its choice, it checks whether or not it needs to "beef up" the chosen class' occupancy of Q's neighborhood. For a threshold R and a top match count m, if $d^j < R$ for all $j \in 1, \cdots, m$, the region is judged sufficiently populated, and no additional training is needed; otherwise, it is underpopulated, and Q is added as a prototype.

As training results in storing new prototypes, old ones must be discarded to avoid accumulating large numbers of them. The system therefore has a limit on the number of prototypes it will store for any one class, and has a scheme for determining which prototype it will discard if it has to learn (store) a new one. The scheme involves keeping a usage count of how many times each prototype was matched, aged (decayed) according to how far in the past the match was (and therefore, how "stale" (no pun intended) that prototype is). The parameters can be tuned to a store's checkout rates and volumes, so that retraining follows closely produce ripening and replenishment cycles. For full details, see [12].

6 Recognition Performance

The system has been extensively tested on a database that contains approximately 5,000 images of some 150 different produce types. These images were obtained from purchases in four different stores; from two other stores, at least one of each available produce item was purchased.

The table in Figure 12 gives an indication of the recognition power of some image features. This test is performed on a database of 48 produce items, with five images of each. In all the images, the items are in the transparent bags commonly found in grocery stores.[4] The system was trained on all images, resulting in a training set $\mathcal{P}$ of 240 (5×48) histograms. Testing was performed by matching the image histograms to these training histograms, while not allowing an image to be classified by matching against itself. That

[4]Type 4 LDPE (Low Density Polyethylene) transparent bags, being more transparent than Type 2 bags, give better classification results.

<table>
<tr><th></th><th>Number 1</th><th>In Top 4</th><th>Number 1</th><th>In Top 4</th><th>Number 1</th><th>In Top 4</th></tr>
<tr><td>Hue</td><td>59%</td><td>87%</td><td rowspan="3">72%</td><td rowspan="3">90%</td><td rowspan="5">80%</td><td rowspan="5">97%</td></tr>
<tr><td>Saturation</td><td>37%</td><td>59%</td></tr>
<tr><td>Intensity</td><td>22%</td><td>55%</td></tr>
<tr><td>Texture A</td><td>24%</td><td>57%</td><td rowspan="2">33%</td><td rowspan="2">63%</td></tr>
<tr><td>Texture B</td><td>23%</td><td>54%</td></tr>
</table>

Figure 12: 48 produce items, five images each.

is, a test histogram Q is classified as $I(P)$ such that $d(Q, P)$, $P \in \mathcal{P}$, is minimal and not zero.

As can be seen from the first two columns of Figure 12, hue is the most powerful color feature, with 59% of the produce items classified correctly based on hue histograms alone; for 87% of the items, the correct classification is in one of the top four choices. Saturation alone does significantly less well, and intensity alone is even worse. The texture measures perform comparably to intensity.

The second two columns of the table of Figure 12 give the results when hue, saturation, and intensity are combined and when two texture features (Texture A and Texture B – see Section 4) are combined. For color (H,S,I) we have that 72% of produce items is classified correctly and 90% the correct choice is in the top four selections. For the combined texture measures, these numbers are 33% and 63%, respectively. Finally, the last two columns give the classification results when color and texture are combined. 80% of the produce items is classified correctly; 97% of the time, the correct answer is in the top four choices. These numbers are typical of the performance we have found throughout all testing.

The above test is on only 48 produce items – about the number of found in a typical small produce department. A larger test of 145 items, every item on the shelf, was performed, using all produce items available in a supermarket.[5] The characteristics of this sample are as follows:

- Most items were sold in bulk; a few were pre-packaged and were unpacked for testing purposes.
- The test set contained many produce items that belong to the same variety, but were of different type or quality, e.g., seven types of apple, seven tomato types.
- Of the 145 items, 50 were mainly green in color (American parsley, Italian parsley, Granny Smith apples, ...) while 95 items were of a different color (bananas, potatoes, ...).
- All the produce items were packaged in transparent plastic bags (similar to bags in the previous test).

Ten images of each item were taken. Here, efforts were made to confuse the system. Items such apples and lemons were imaged with a varying number of them in the bag. When possible, items were photographed in odd positions, including some that are not likely to occur at a real checkout counter, for example, a broccoli balanced on its head. Items that have non-uniform appearance, such as carrots with the leaves attached, were imaged with the different (e.g., orange and green) parts exposed in different images. Such imaging could possibly adversely effect the recognition results.

[5] Food Emporium, Rye, NY (mid-winter).

	Number 1	*In Top 4*
Color	79%	93%
Color and Texture	84%	96%

Figure 13: Every item on the shelf, ten images each.

Testing of this data set was done using leave-one-out [13]. Or, to be more precise, the system was trained on all 1,450 images and then tested on the images. Whenever a classification distance $d(Q, P)$, with Q the test histogram and P a prototype histogram, was zero, the second best prototypes identifier was selected as the class class of Q. Figure 13 gives the results of using color alone and color combined with texture. The results are very much in agreement with, and even a little better than, those given in Figure 12. For color and texture combined, 84% of the time the correct produce class was selected, 96% of the time, the correct class was present in the top four choices. The results are better than those reported for the smaller test set above because the segmentation algorithms were more refined at the time of the test.

The table of Figure 14 gives some results of experiments with different training and learning techniques. The database consists of 506 images of 51 produce types (10 images of most produce types). Only color is used in these experiments.

The first row of Figure 14 gives the normal matcher results. All images are used for training and all images are classified using these prototypes, but if an image is matched to itself, the second choice is selected. 95.1% is classified correctly in the first four choices, and for 80.8%, the top choice is the correct one.

The second row reports results in which, for each produce item, the first five images

Case	*Number 1*		*In Top 4*	
Normal matcher	80.8%		95.1%	
Normal matcher, 5-5	79.3%	-1.5	94.0%	-1.1
Automatic selection	78.7%	-2.1	94.1%	-1.0
5" × 5" window	78.7%	-2.1	91.5%	-3.6

Figure 14: Comparison of some techniques to base matching technique.

were used for training and second five images were used for testing. Classification degraded by slightly more than 1%.

In the row labeled "Automatic selection," training instances are automatically selected, based on distances in the feature space. A new training sample P' of a class is only added as a prototype of that class if $d(P', P) > T_{Add}$, for all prototypes P in that class. The result is that, on average, 6.4 prototypes are stored per class. Classification results degrade a little, but not enough to be considered statistically significant.

The last row simulates the row one experiment, but as though the size of the enclosure's glass plate had been reduced to 5" × 5", roughly the size of window that is found in today's embedded barcode readers. Only the central region of the data set images was processed. Some degradation of performance occurs, but it cannot be considered alarming.

Human selection of the correct produce class for every produce item is considered undesirable. It is preferred that the system give *only* one choice if the top choice is in some sense reliable. Such "hard" or forced classification is achieved by emitting only one classification if the two top matching classes are the same. Figure 15 gives some results. The same data set as in Figure 14 – 506 images of 51 vegetable types – is used. The first row gives the normal matcher results: 69.8% of the time, the system's top choice is the correct classification, 11% worse than the normal matcher of Figure 14. 92.3% of the time, one of the up to four classes displayed is the correct one. (Sometimes fewer than four are displayed.) Forcing a single choice has its drawbacks; 4.15% of the time this choice is wrong (false positive), and in about 2/3 of these cases (2.77% of the time), the correct choice was among the other choices whose display was suppressed. (These latter numbers are indicated in colums labeled *FalsePositive* and *CorrectIn4*.) An 11% drop is indicated (compared to row one, Figure 14) in this table. Note that this is due to those items that have an item of the same class as nearest neighbor and one of different class as second nearest neighbor, which would have been classified as top choice in Figure 14.

The second row in the table of Figure 15 gives the results of the same experiment when automatic selection (as in Figure 14) is used. This leads to a 23.1% drop in the top choice selection, which can be attributed to the fact that fewer prototypes per class are available (6.4 per class, versus 9 per class), and they are more spread apart. (They will roughly span the same volume of feature space, though.)

Case	*Correct*		*In Up To 4*		*FalsePositive*	*CorrectIn4*
Normal matcher	69.8%	-11.0	92.3%	-2.8	4.15%	2.77%
Automatic selection	57.7%	-23.1	90.5%	-4.6	4.55%	3.56%

Figure 15: Comparison of some techniques to base matching technique.

It is interesting to examine the recognition performance on produce when the training data is completely unrelated to the test set. This is exactly what is displayed in Figure 16 (color only). Three sets of produce are used: 145 items purchased from a Turcos

		Reference		
		Turcos	*Food Emporium 1*	*Food Emporium 2*
Test	*Turcos*	77.7 / 92.3 (145)	38.0 / 68.2 (97)	33.7 / 65.6 (75)
	Food Emp. 1	34.1 / 62.5 (96)	85.8 / 96.7 (116)	39.9 / 68.5 (82)
	Food Emp. 2	24.6 / 56.1 (73)	31.4 / 67.6 (79)	84.4 / 97.5 (89)

Figure 16: Cross data set testing.

supermarket in January, 116 items from a Food Emporium, purchased in February, and 89 items bought from a different Food Emporium in March. Hence, the items were bought from different stores and in different months (but all in the winter). Two of the stores are in the same chain; the third store is independent. Other than that, not much is known about the data sets, i.e., the region of origin, the wholesaler, the methods of storage, all could be different for the two sets.

The cells in the table refer to tests that are performed with the system trained on a particular set. The first number gives the percentage of correctly classified items, the second one gives the percentage that is classified in the top four. This ranges roughly from 25% / 56% to 40% / 68%. These are the type of results that can be expected when the system is trained in one region in one season and then tested in a different region at a different time. Incremental training as discussed in Section 5 will quickly fine-tune the performance.

The number in parentheses in Figure 16 indicates the number of items that two data sets have in common. There is not much consistency in the naming conventions of produce in the various stores ("Indian River Red Grapefruit"/"Pink Grapefruit"). Items with the same name in different stores are not necessarily the same item ("baking potato").

Same Store Summary

	Mean	Std. Dev.
Top Choice	82.6%	3.5
In Top 4	95.5%	2.3
Classes Tested	117	23

Cross Store Summary

	Mean	Std. Dev.
Top Choice	33.6%	4.9
In Top 4	64.8%	4.4
Classes Tested	84	9

Figure 17: Summary of multiple store classification results.

The table of Figure 17 summarizes the results. Here, the average and standard deviation of the results of Figure 16 are given. On the left, the average of training and testing on same-store data is given: 82.6% top choice, 95.5% in top four. The standard deviation is rather small, indicating that these types of results can be expected regardless of the store. The table on the right in Figure 17 gives the performance numbers that are obtained when the system is trained on a data set that is completely unrelated to the data that it is tested on.

7 Discussion

Automatic visual recognition of produce – either at the point of sale or in the produce department – is not as unattainable a goal, as is commonly believed. We have developed a produce recognition system that uses a color camera to image the items. A special purpose imaging setup with controlled lighting allows very precise segmentation of the produce item from the background. From these segmented images, recognition clues such as color and texture are extracted.

Of course, just as a human cashier, our system does not achieve 100% correct recognition. From the outset, the user interface has been designed with this in mind. Only when the classification of a produce item is judged to be very reliable, one top selection produce class is given by the system; in all other cases, the human operator is asked to make the final classification by selecting from a displayed set of produce images. When using *only* color as classification cue, roughly the following classification results are obtained:

60%	sure and generates the correct answer
30%	puts up multiple choices, one of which is correct
5%	sure but guesses wrong (false positive)
5%	puts up multiple choices, none of which is correct

Currently, color and texture are the best developed features, and the ones that contribute most to reliable recognition. Of these, color is by far the more important feature. Using color alone, quite respectable classification results are achieved. Adding texture improves the results, in the range of 5 - 10%. Features such as shape and size can augment the feature set to improve classification results. However, incorporation of these features slows down the recognition time, which is currently under 1 second on special purpose hardware on a PC based machine.

8 Acknowledgement

We would like to thank Thomas Cofino, Mario Schkolnick, and Kevin McAuliffe for all their support and encouragement during the development of this system. IBM Raleigh (R&DS) is thanked for their financial support. Louis Percello was instrumental in the filing of the patent disclosures. Shree Nayar is thanked for his helpful hints.

References

[1] R. Petreycik. Produce. *Supermarket Business Magazine*, 47(9):129, September 1992.

[2] S. Mersch. Polarizing lighting for machine vision applications. In *RI/SME Third Annual Applied Machine Vision Conference*, pages 40–54. Schaumburg, February 1984.

[3] S. K. Nayar, X. Fang, and T. E. Boult. Separation of reflection components using color and polarization. Technical Report CUCS-058-92, Columbia University, 1994.

[4] M. J. Swain and D. H. Ballard. Color indexing. *Int. Journal of Computer Vision*, 7(1):11–32, 1991.

[5] D. H. Ballard and C. M. Brown. *Computer Vision.* Prentice Hall, 1982.

[6] R. W. G. Hunt. *Measuring Color (Second Edition).* Ellis Horwood, West Sussex, England, 1991.

[7] M. Miyahara and Y. Yoshida. Mathematical transform of (R, G, B) color data to Munsell (H, V, C) color data. In *SPIE Vol. 1001 Visual Communications and Image Processing*, 1988.

[8] R. Haralick. Statistical and structural approaches to texture. *Proc. of the IEEE*, 67(5):786–804, May 1979.

[9] L. van Gool, P. Dewaele, and A. Oosterlinck. Texture analysis anno 1983. *Computer Vision, Graphics, and Image Processing*, 29:336–357, 1985.

[10] H. Tamura, S. Mori, and T. Yamawaki. Texture features corresponding to visual perception. *IEEE Trans. on Systems, Man, and Cybernetics*, 8(6):460–473, 1978.

[11] K. I. Laws. *Textured image segmentation.* PhD thesis, Univ. Southern California, 1980.

[12] R.M. Bolle, J.H. Connell, N. Haas, R. Mohan, and G. Taubin. Veggievision: A produce recognition system. Technical Report forthcoming, 1996.

[13] A. K. Jain and R. C. Dubes. *Algorithms for clustering data.* Prentice Hall, Englewood Cliffs, New Jersey, 1988.