Proceedings of the 2005 IEEE/ASME
International Conference on Advanced Intelligent Mechatronics
Monterey, California, USA, 24-28 July, 2005

Effects of Color Characterization on Computational Efficiency of Feature Detection with Live-object Handling Applications

Qiang Li* and Kok-Meng Lee

Abstract— **This paper presents a machine vision algorithm that utilizes the principal component analysis technique to characterize target features in color space from a set of training data so that the color classification can be done accurately and efficiently. The method, referred to here as the statistically based fast bounded box (SFBB), has significant potential in agriculture and food processing applications where color variability often renders grayscale-based algorithms difficult or impossible to work. We evaluate the algorithm in the context of live-bird handling applications and examine the effects of the color characterization on computational efficiency by comparing the proposed solution against two commonly used color classification algorithms; the RCE neural network classifier and the support vector machine. Comparison among the three methods demonstrates that SFBB is relatively easy to train, efficient and effective since with sufficient training data it requires no additional optimization steps; these advantages make SFBB an ideal candidate for high-speed automation involving live and/or natural objects.**

Keywords: **Machine Vision, Inspection, Feature Detection, Color classification**

I. INTRODUCTION

Natural object identification has received more and more attention in automation. Early work focused on identifying a human face from grayscale images [1-4] using edge and shape information. More recently, color vision as an image processing tool in detecting features has widely been adapted to human face identification [5-8]. The algorithm based on color has been found to be much faster than those based on shape. However, color vision has its problems when used in natural object identification since no two individuals are identical. In addition, perceived color can be very different under different lighting conditions. Design of a time-efficient, reliable color classification algorithm for natural, live object applications has remained a challenge.

Machine vision (MV) algorithms for detecting features of a moving natural-object can be classified into three main categories: (1) Features are extracted from a gray-level image on the basis of both edge and shape. (2) Features are detected by virtue of their characteristic color. (3) The method uses a combination of (1) and (2); often characteristic color detection is applied as a pre-processor followed by a shape matching algorithm to identify the target feature. Among them, characteristic color detection has been an attractive solution particularly in applications where the color difference between the target features and its background is significant. For high-speed automation applications such as handling of live objects for meat production, the shape variation and voluntary motion of live, natural objects coupled with the stringent production demands to reduce computation time make algorithms based on shape information look less than attractive for real-time applications. The characteristic color detection algorithm (CCDA) is especially suitable for detecting features of live natural objects and has significant potential in agriculture and food processing. However, color images of natural product are susceptible to noise as good lighting system can only partially solve the noise problem. In addition, color variation and uniformity are common; a unique nature of live objects. To improve the success rate of detecting color features involving live objects, the CCDA must have the ability to discriminate target features from color noise at high speed. For these reasons, our color vision research has focused on addressing two specific issues: The first is to create artificial color contrast that aims at highlighting the target while suppressing its surrounding; this thrust has been reported in [9]. The second, *the thrust of this paper*, is to improve the characterization of the feature color in an attempt to exclude noise so that the color classification can be done more accurately and efficiently.

The basic idea of a CCDA is to utilize a set of training data to approximate the boundary of the color subspace that characterizes the feature for subsequent classification. RCE neural network classifier (RCE-NNC) [10] has been one of the most commonly used methods for identifying shape patterns [3], [11-15]. It has also been used in color vision [16] and orientation detection in handling live birds [17]. The performance of RCE-NNC as compared with other neural networks has been studied in [18]. The success rate of a RCE-NNC depends heavily on its designed parameters and the topology of the trained network. Although some research effort (for example [13] [19]) has been directed towards optimizing the topology of a RCE-NNC, its optimization for a real-time application has remained a challenge. More recently, kernel methods have become more and more popular [20]. As one of the commonly used kernel methods, the support vector machine (SVM) [21-23] has been developed from rigorous statistical learning theory. Both RCE-NNC and SVM are relatively easy to use as compared to

Manuscript received March 1, 2005. This work was supported in part by the Georgia Agricultural Technology Research Program (ATRP) and the U.S. Poultry and Eggs (USPE) Association.

*Corresponding author, Q. Li is a Ph.D. Candidate in the Woodruff School of Mechanical Engineering at the Georgia Institute of Technology, Atlanta, Georgia 30332-0405, USA (e-mail: gtg873k@mail.gatech.edu)

, K.-M. Lee is a Professor of the Woodruff School of Mechanical Engineering at the Georgia Institute of Technology, Atlanta, Georgia 30332-0405, USA (e-mail: kokmeng.lee@me.gatech.edu)

many other NNC's. In practice, both methods however could become less efficient and inaccurate when their parameters are poorly tuned; the procedure for optimizing the design parameters could be tedious and time-consuming.

The remainder of this paper offers the following:

1) We present an alternative method, SFBB, to characterize the feature color. The SFBB uses the principal component analysis (PCA) technique [24] to obtain the principal axes of the training data distribution in the color space. The principal component analysis is well known for its use in the eigenface algorithm [1] [2] that helps find the most dominant feature from a grayscale image of a human face. Unlike [1] [2] where PCA was used to reduce the dimension of the identification problem with the whole grayscale image as a classification input, the SFBB method finds a linear transformation to minimize the covariance of the training set where individual color pixels are used as classification inputs.

2) We illustrate the use of SFBB for color characterization of features in the context of an automaton problem, where reliability and high processing speed are of particular concern. Specifically, we apply the SFBB to a live-object handling application where variability in natural objects is usually several orders-of-magnitude higher than that for manufactured goods.

3) We examine the effects of the color boundary on the computational efficiency of feature detection by comparing the three color classification algorithms; namely, the SFBB, the RCE neural network classifier (RCE-NNC), and the support vector machine (SVM). The SFBB uses a bounded box with orientation and dimensions defined by the statistics of the training set while the RCE-NNC uses hyper-spheres and the SVM uses hyper-planes.

II. STATISTICALLY-BASED FAST BOUNDED BOX (SFBB)

A color signal is represented here in RGB (red, green and blue) space, and we consider here the target as a subspace in the entire RGB color space.

A. Problem Formulation

The identification problem can be defined as follows: *Given a set of scatter points (referred to here as a training set) in the target color subspace Ω, the problem is to find its boundary Γ such that for any color vector C in the color space, if C is inside Γ then C represents one point on the target feature; otherwise it does not belong to the feature.*

If all the members in Ω are known, the boundary of Ω is also known. Unfortunately, the boundary of the target color subspace can only be constructed from a limited set of training samples; thus, it is essentially an approximation at best! The closeness and shape of the approximated boundary, which depends on the decision rules of the specific CCDA employed, have a significant influence on the cycle time and success rate of the CCDA performance.

B. Simple Bound Box in RGB Space and its Problems

The interest here is to describe the boundary Γ that simplifies the subsequent identification process and that makes CCDA *accurate* and *fast*. A simple bounded box can be used to approximate the boundary of the target color subspace. *This method assumes that the three color component vectors are independent random variables.* The basic idea here is to construct a smallest possible rectangular box to enclose representative color points of the target from a given training set in RGB space. Such a bounded box can be easy constructed from the maximum and minimum values of the color component. The classification can then be reduced to simply checking whether the RGB pixel values are within the bounds, which is a relatively simple and fast process.

One problem with the above procedure is that it does not result in a tightest box since it does not take into account the color characteristics of the feature. In addition, this larger-than-necessary box would result in introducing unwanted color points (as noise) in the processed image.

A relatively simple method to examine whether the three color component vectors are independent random variables is to fit the normal distribution to the training data. If the training set matches the ideal normal distribution very well, there is a very high probability that these components are independent of each other. Otherwise, some of them may be related and a change of the variables is necessary to minimize correlation among the variables.

B. Finding SFBB using Principal Component Analysis

In order to minimize the correlation about the component vectors, the SFBB method computes the three principal axes of the training set for constructing the bounded box. The method involves the following steps.

- *First, the principal axes characterizing the training set are calculated from its covariance matrix.*

Given a training set **R** with three color component vectors $\mathbf{R_r}$, $\mathbf{R_g}$ and $\mathbf{R_b}$ in the RGB space, a covariance matrix [**U**] can be computed from (1):

$$\left[\mathbf{U}\right]=\begin{bmatrix}\sigma^2\{\mathbf{R_r}\} & \sigma\{\mathbf{R_r},\mathbf{R_g}\} & \sigma\{\mathbf{R_r},\mathbf{R_b}\}\\ \sigma\{\mathbf{R_g},\mathbf{R_r}\} & \sigma^2\{\mathbf{R_g}\} & \sigma\{\mathbf{R_g},\mathbf{R_b}\}\\ \sigma\{\mathbf{R_b},\mathbf{R_r}\} & \sigma\{\mathbf{R_b},\mathbf{R_g}\} & \sigma^2\{\mathbf{R_b}\}\end{bmatrix} \tag{1}$$

where $$\sigma^2\{\mathbf{R_i}\}=\frac{1}{N-1}\sum_{n=1}^{N}\left(R_{in}-\bar{R}_i\right)^2 \qquad i\in\{r,g,b\} \tag{1a}$$

$$\sigma\{\mathbf{R_i},\mathbf{R_j}\}=\frac{1}{N-1}\sum_{n=1}^{N}\left(R_{in}-\bar{R}_i\right)\left(R_{jn}-\bar{R}_j\right) \qquad i,j\in\{r,g,b\};i\neq j \tag{1b}$$

$$\bar{R}_i=\frac{1}{N}\sum_{n=1}^{N}R_{in} \tag{1c}$$

and N is the number of color points. The covariance matrix [**U**] is symmetric. To maximize the variances of interest, we apply the singular-value- decomposition method to obtain the eigenvalues (λ_1, λ_2, λ_3) and the normalized eigenvectors ($\mathbf{v_1}$, $\mathbf{v_2}$, $\mathbf{v_3}$) of [**U**]. The three principal axes of the new coordinate system are given by the unit vectors along $\mathbf{v_1}$, $\mathbf{v_2}$, and $\mathbf{v_3}$.

- *Second, the training set in RGB space is then transformed to the new coordinate system aligned with the three principal axes of the training set.*

The training data can be transformed from its original RGB space to the new coordinate system:

$$\hat{\mathbf{R}} = \begin{bmatrix}\mathbf{v}_1 & \mathbf{v}_2 & \mathbf{v}_3\end{bmatrix}^T (\mathbf{R} - \bar{\mathbf{R}}) \tag{2}$$

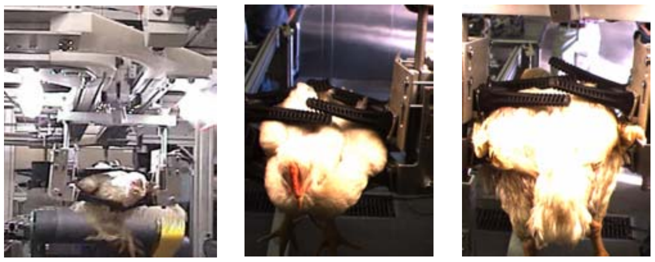
where $\mathbf{R} = \begin{bmatrix}R_r & R_g & R_b\end{bmatrix}^T$ and $\bar{\mathbf{R}}$ are the original vector and its corresponding average in RGB space respectively; and $\hat{\mathbf{R}} = \begin{bmatrix}R_1 & R_2 & R_3\end{bmatrix}^T$ is the transformed color vector.

- *Finally, the bounded box is constructed in the new coordinate system.*

In order to find an appropriate size for the bounded box to best characterize the color of the feature, we use linear regression theory to determine the confidence level at which the three transformed color components are independent random variables; this level is used to specify the boundary of the box in terms of standard deviations (SD).

III. Live-bird Handling Application

We illustrate the computation procedure with a live-bird handling application, where the birds must be shackled in a specific direction. The bird's orientation (forward/backward) is determined by identifying its red comb relative to its white-feathered body as shown in Fig. 1. Due to varying sizes and shapes and some natural reflexes (or voluntarily motion) of the birds, machine vision algorithms based on edge and/or shapes have difficulties meeting stringent production requirements that demand reliability and speed. In this application, the vision algorithm is used to detect the red comb of a bird. Apart from a spectrum of red that characterizes the combs in a typical batch, noise (such as dirt on the feathers, bare spots of flesh, shadow and reflection of environmental illumination) present a challenge to reliable color detection.

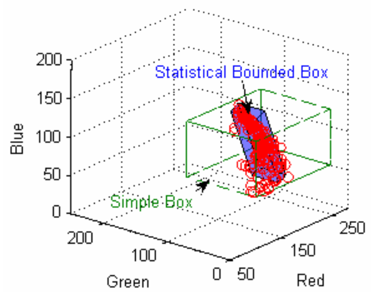


(a) Setup (b) Forward (c) Backward

Fig. 1 Experimental setup and typical images

Figure 2 shows a set of experimentally obtained training data of *N* color points from the comb. The data are plotted in RGB space, along with its corresponding covariance matrix and normalized eigenvectors, where the rectangular box represents the computed SFBB (with dimensions at ±2 SD or 95% confidence level). Figure 3 compares the normal distribution of the red components of the training set against those of the transformed vector components. In Fig. 3, the probability distribution is graphed in natural logarithmic scale; the dashed line is the ideal normal distribution and the discrete points are experimentally obtained data.

As shown in Fig. 3, the original red component data do not match the normal distribution and thus, its vector is not an independent random variable. The data distribution in transformed coordinates matches the normal distribution very well; the transformed coordinates can be treated as independent random variables. This justification permits the boundary of the bounded box to be specified in terms of the number of standard deviations (SD's) of each component. Clearly, the tightest box (that more closely characterizes the color of the comb) is preferred since a larger-than-necessary box would include un-related features as noises, which must be excluded before color classification. Once the SFBB is obtained, the classification becomes a straight-forward process; any color pixels that fall inside the box will be considered as feature color and those pixels that appear outside of the box are considered as noise pixels.



Mean:

$\bar{R} = (242, 112, 69)$

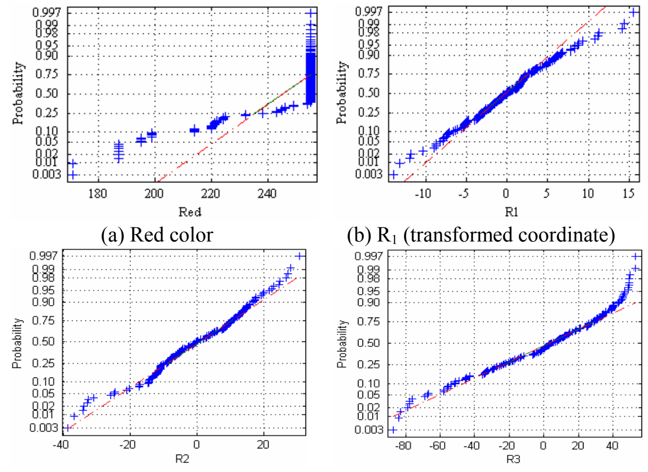
Covariance:

$$[\mathbf{U}] = \begin{bmatrix} 441 & 384 & 226 \\ 384 & 666 & 432 \\ 226 & 432 & 325 \end{bmatrix}$$

Eigenvectors:

$$\begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \mathbf{v}_3^T \end{bmatrix} = \begin{bmatrix} -.14 & .61 & -.78 \\ .86 & -.32 & -.41 \\ .50 & .72 & .48 \end{bmatrix}$$

Fig.2 Training set and the computed SFBB, data from Fig. 1(b).



(a) Red color (b) R_1 (transformed coordinate)

(c) R_2 (transformed coordinate) (d) R_3 (transformed coordinate)

Fig. 3 Normal distribution (before and after transform)

IV. Effects of Color Characterization

We examine the effects of the target-subspace boundary for characterizing the target color by comparing the SFBB against two other methods; RCE neural network classifier (RCE-NNC) and support vector machine (SVM).

A. RCE Neural Network

A three-layer RCE-NNC (shown in Fig. 4) is used to provide supervised learning of color pattern categories separated by *nonlinear, essentially arbitrary boundaries*. The concept of a pattern class develops from storing in memory a limited number of class elements (prototypes). Associated with each prototype is a modifiable scalar weighting factor (λ) that defines the threshold for categorizing an input to the prototype. Learning involves (1) commitment of the prototypes to memory and (2) adjustment of the various λ factors to eliminate classification errors.

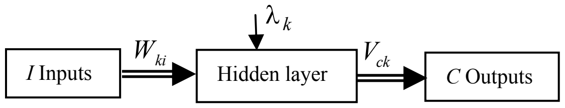


Figure 4 Structure of the trained network

The three-layer RCE-NNC has *I* input nodes (equal to the dimension of input vector), *C* output nodes for the number of output categories, and a hidden layer. The hidden layer is initially empty and creates nodes dynamically through learning. If the new pattern does not belong to an existing class (or is not within the sphere defined by λ_k), a new node is created in the hidden layer. A default threshold λ and a distance function $D(\mathbf{v}_i, \mathbf{v}_j)$ where $\mathbf{v}_i$ and $\mathbf{v}_j$ are two vectors in the color space must be assigned before training can begin, which could significantly influence the number of nodes generated in the hidden layer and the type of hidden node (for example, sphere or rectangular box) respectively. Thus, these design parameters determine the topology of trained network, and hence the performance of classifier. In this study, the distance function is a 3D Euclidean distance between two vectors in the color space computed by:

$$D(\mathbf{W}_k, \mathbf{R}_n) = \sqrt{(r_k - r_n)^2 + (g_k - g_n)^2 + (b_k - b_n)^2} \tag{3}$$

The training process of a RCE-NNC is iterative. We illustrate here using the following pseudo-code:

```
Initialization
  Number of nodes on the hidden layer: M =0
Training begins
  Let 1st pattern $\mathbf{R}_1$ belong to c th class resulting in 1st hidden node, M=1
      $W_{1i} = R_{1i}$ ; $V_{c1}$ =1; and $\lambda_1$=default threshold, where $i = r, g, b$.
  The process repeats and new hidden cells are created.
Confusion
  With M cells available, the following process repeats for each new training
      pattern. Consider nth training pattern (belonging to the dth class) arrives.
    For $k=1: M$
      $m_k = N_{\lambda k}\left(D(\mathbf{W}_k, \mathbf{R}_n)\right)$
      where $D(\mathbf{W}_k, \mathbf{R}_n) = \|\mathbf{W}_k - \mathbf{R}_n\|$ and $N_{\lambda k}(x) = \begin{cases} 1, & x < \lambda_k \\ 0, & x \ge \lambda_k \end{cases}$
        If $m_k = 1$ and $V_{dk} = 1$
            Pattern is correctly contained within a cell.
        Else if $m_k = 1$ and $V_{ck} = 1$
            The classification is incorrect. Decrease $\lambda_k$ until $m_k = 0$
        End
        If $\forall k$ , $m_k = 0$ then add new cell in the hidden layer.
            $\mathbf{W}_{M+1} = \mathbf{R}_n$
            $V_{d(M+1)} = 1$, and
            $M = M + 1$
        End
    End
```

Once it is trained, the RCE-NNC stores the points in a metric space R^N. The boundary of the class is approximated by a set of hyper-sphere in feature metric space. The distance function relates the unknown pattern to a category. The weight of the hidden node is the coordinate of the center of the hypersphere. The threshold of hidden node represents the radius of the hypersphere. A vector in metric feature space will be recognized as one class if it falls into one of the hyperspheres that belongs to that class.

B. Support Vector Machine (SVM)

The SVM, a classification method developed from linear statistical classifier, is the combination of hyper-plane classifier [25] and kernel method [26] [27] and is briefly introduced here for completeness. Given the *training set*

$$S = \{(\mathbf{x}_1, y_1) \ldots, (\mathbf{x}_l, y_l)\}$$

where $\mathbf{x}_i \in X \subset \Re^N$ and $y_i \in Y = \{1, \ldots, k\}$ are the predictive (or independent) variable and the target (or dependent variable), we wish to obtain a mapping:

$$f_S : X \to Y$$

The SVM discussed here is a two-class classification problem, the classes being *P, N* for $y_i = \pm 1$, which can easily be extended to *k* class classification by constructing *k* two-class classifiers.

Maximal Margin Hyper-planes

If the training data are linearly separable then there exists a pair $(\mathbf{w}, b)$ such that

$$\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \begin{cases} \ge +1, & \text{for all } \mathbf{x}_i \in P \\ \le -1, & \text{for all } \mathbf{x}_i \in N \end{cases} \tag{4}$$

The decision rule is given by

$$f_{\mathbf{w},b}(\mathbf{x}) = \operatorname{sgn}(\mathbf{w}^T \mathbf{x} + b) \tag{5}$$

where $\mathbf{w}$ is the weight vector; and b is the threshold. As shown in the appendix, the optimal linear hyper-plane is constructed in the feature space by applying the margin principle which will maximize the margin between two classes. The decision function is then given by

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \lambda_i^* \mathbf{x}^T \mathbf{x}_i + b^*\right) \tag{6}$$

where $\mathbf{w}^* = \sum_{i=1}^{l} \lambda_i^* y_i \mathbf{x}_i$; $b^* = y_i - \mathbf{w}^{*T}\mathbf{x}_i$; and $\Lambda = (\lambda_1, \ldots, \lambda_l)^{\mathrm{T}}$ are the Lagrange multipliers to be obtained by solving the following dual problem given by (A6) in the appendix

$$\text{Maximize} \quad F(\Lambda) = \Lambda^{\mathrm{T}} I - \tfrac{1}{2}\Lambda^{\mathrm{T}} D \Lambda \tag{7}$$

subject to $\Lambda \ge 0$, $\Lambda^{\mathrm{T}} y = 0$; *D* is a symmetric $l \times l$ matrix with elements

$$D_{ij} = y_i y_j \mathbf{x}_i^T \mathbf{x}_j \tag{8}$$

Note that Λ is only non-zero when $y_i(\mathbf{w}^T \mathbf{x}_i + b) = 1$, vectors for which they are called *support vectors* since they lie closest to the separating hyper-plane.

The solution obtained is often sparse since only $\mathbf{x}_i$ with non-zero Lagrange multipliers appear in the solution. This is important when the data to be classified are very large, as is often the case in practical classification situations. However, it is possible that the expansion includes a large proportion of the training data, which leads to a model that is expensive both to store and to evaluate; alleviating this problem is an area of ongoing research in SVM.

Kernel Feature Spaces

A linear classifier may not be the most suitable hypothesis for the two classes. The SVM can be used to learn non-linear decision functions by mapping the data to some higher dimensional *feature space* and constructing a separating hyperplane in this space. Denoting the mapping by

$$X \to H \quad \text{and} \quad \mathbf{x} \mapsto \phi(\mathbf{x}) \tag{9}$$

Mapping data to H is however time consuming and storing it may be impossible. Note that H could be infinite dimensional. Hence a *kernel function*

$$K(\mathbf{x},\mathbf{z}) \equiv \phi(\mathbf{x})^{\mathrm{T}} \phi(\mathbf{z}) \tag{10}$$

is introduced and the decision function becomes

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \lambda_i^* K(\mathbf{x},\mathbf{x}_i) + b^* \right) \tag{11}$$

where the bias for any support vector $\mathbf{x}_i$ is given by

$$b^* = y_i - \sum_{j=1}^{l} y_j \lambda_i^* K(\mathbf{x}_j,\mathbf{x}_i) \tag{12}$$

The kernel function allows us to construct an optimal separating hyperplane in the space H without explicitly performing calculations in this space, which requires K to be an easily computable function; commonly used kernels include linear, polynomial and radial kernels. In this study, we choose a radial basis function for the kernel:

$$K(\mathbf{x}_i,\mathbf{x}_j) = \exp(-\gamma \left\| \mathbf{x}_i - \mathbf{x}_j \right\|^2), \gamma > 0 \tag{13}$$

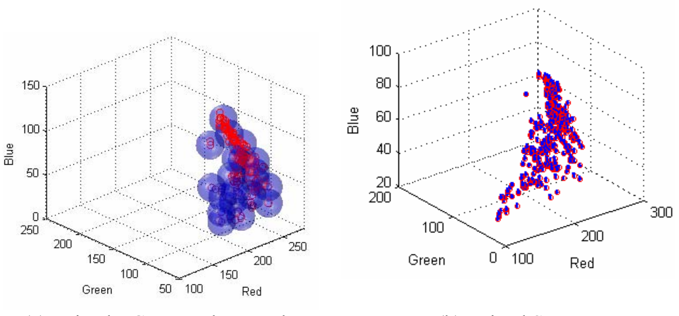
The SVM is reduced to solving the Lagrange multipliers from (7) with $D_{ij} = y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)$, the solution of which can be obtained using quadratic programming techniques.

C. Comparison of Results

We examine experimentally the effects of the boundary approximation on computational efficiency in color space by comparing the SFBB against the RCE-NNC and SVM. The studies were performed in the context of an automation problem and orientation detection of a live bird as described in Section III. Eighty one (640x480-pixel) images of grasped birds were captured; and divided into two groups, 51 birds facing forward and the remaining 30 facing backward. The training sets are the same as in Figs. 1 and 2. The images were pre-processed with a difference-of-Gaussian filter [9] to enhance contrast, and post-processed with a morphological operation based on the majority rule of 10pixels (in an 8x8 mask) to remove isolated noises.

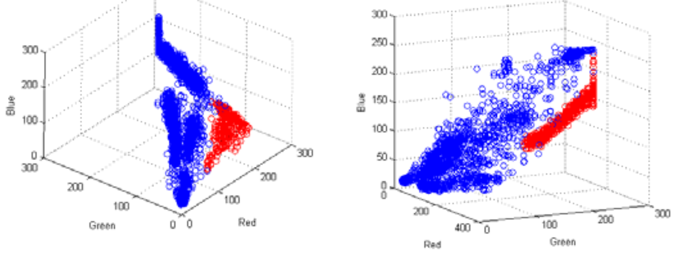
The following summarize the differences among the three methods and observations made from the results:

1) The SFBB uses a bounded rectangular box (Fig. 2) with its orientation and dimensions defined by the training set statistics, while the RCE-NNC and SVM use hyper-spheres and hyper-planes respectively. To offer a visual comparison, Fig. 5(a) shows the geometry topology of trained RCE neural network where we choose λ=8, minimum threshold =1, and the distance function defined in Equation (2). The geometry of the generated nodes of a REC-NNC in the hidden layer is spherical in 3D space. Fig 5 (b) shows the boundary of the same training set but using the SVM classifier. Due to the difficulty in solving the boundary of the SVM in closed form, the boundary was numerically computed by testing each of the pixel cells in RGB space using the SVM classifier. The connected cells are combined to form the boundary. The boundary so generated is not smooth; however, it is intuitive in understanding the SVM classifier boundary.



(a) trained RCE neural network (b) trained SVM

2) While the SVM is a more general method for solving classification problems, it has two major disadvantages:
   a) Unlike SFBB and RCE-NNC which can construct the boundary for a single class, SVM needs at least two classes to construct a hyper-plane. Although the primary target features are the *red comb* and *white body* (as background), there are points in the image that are neither body nor comb. Thus four classes are needed; (B/NB) – feature color for body and not body, and (C/NC) – for comb and not comb for this problem.



(a) C and NC (b) B and NB

Fig. 6 Training data of SVM

   b) The performance is very sensitive to the training data. The training patterns for non-target features must be carefully chosen in the proximity of the target in color space so that the boundary of the color subspace is tight.

3) The classification results are summarized in Table 1.

Table 1: Classification results.

	Forward	Backward	All	% S Rate	Time
SFBB	50/51	30/30	80/81	99%	T
RCE-NNC	51/51	18/30	69/81	85%	1.9T
SVM	30/51	30/30	60/81	74%	13T

T=5.8 s/image (MATLAB) and T=0.5s /image (C++)

As shown in Table 1, the SFBB correctly identify 80 of 81 cases. The failed case corresponds to an image of a female bird with a small/pale comb, which was captured slightly off timing and thus in dim illumination. Written in C++ code, the average cycle time is about 0.5 second per image. RCE-NNC performed poorly in detecting backward facing birds as it introduces excessive noise. On the other hand, SVM missed a number of red combs as the boundary of the color subspace is rather tightly fitted. The SFBB appears to have the best potential to meet both cycle time and reliability requirements. In addition, the boundary approximation of a SFBB is relatively straight forward and relies only on the standard deviation of the training data distribution to specify its bounds.

V. Conclusion

A new algorithm has been introduced for characterizing feature color of natural, live objects. We also examine experimentally the effects of the boundary approximation of the color characterization on computational efficiency. As compared against two other commonly used methods (the RCE-NNC and the SVM), this algorithm has several advantages including simplicity in training, and fast classification since only three simple checks of rectangular bounds are performed. The computational efficiency and reliability of all three methods on color classification have also been evaluated in the context of an automation problem. The study shows that the statistically based fast bounded box can satisfy the stringent requirements of live bid handling automation.

Appendix

The inequality constraints (4) can be combined to give

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \text{for all } \mathbf{x}_i \in P \cup N \tag{A1}$$

The pair ($\mathbf{w}$, b) can be rescaled such that $\min\limits_{i=1,\dots,l} \left|\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b\right| = 1$, The learning problem is hence reformulated as minimize $\|\mathbf{w}\|^2 = \mathbf{w}^{\mathrm{T}}\mathbf{w}$ subject to the constraints of linear separability (A1). This is equivalent to maximizing the distance (or the normal to the hyperplane) between the convex hulls of the two classes; this distance is called the margin. The optimization is now a convex quadratic programming (QP) problem

$$\underset{\mathbf{w},b}{\text{Minimize}}\ \Phi(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2, \quad \text{subject to } y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1, \quad i = 1,\dots,l.$$

This problem has a global optimum and thus avoids the problem of many local optima in NN training. The Lagrangian for this problem is

$$L(\mathbf{w},b,\Lambda) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{l} \lambda_i \left[y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) - 1 \right] \tag{A2}$$

where $\Lambda = (\lambda_1,\dots,\lambda_l)^{\mathrm{T}}$ are the Lagrange multipliers, one for each data point. The solution to this quadratic programming problem is given by maximizing L with respect to $\Lambda \geq 0$ and minimizing with respect to $\mathbf{w}$, b. Differentiating with respect to $\mathbf{w}$ and b and setting the derivatives equal to 0 yields

$$\frac{\partial L(\mathbf{w},b,\Lambda)}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{l} \lambda_i y_i \mathbf{x}_i = 0 \ \text{ and. } \frac{\partial L(\mathbf{w},b,\Lambda)}{\partial b} = -\sum_{i=1}^{l} \lambda_i y_i = 0 \tag{A3}$$

So that the optimal solution is given by (6) with weight vector

$$\mathbf{w}^* = \sum_{i=1}^{l} \lambda_i^* y_i \mathbf{x}_i \tag{A4}$$

Substituting (A3) and (A4) into (A2) we can write

$$F(\Lambda) = \sum_{i=1}^{l} \lambda_i - \frac{1}{2}\|\mathbf{w}\|^2 = \sum_{i=1}^{l} \lambda_i - \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j \tag{A5}$$

which can be written in matrix notation:

$$F(\Lambda) = \Lambda^{\mathrm{T}} I - \frac{1}{2}\Lambda^{\mathrm{T}} D \Lambda \tag{A6}$$

where D is a symmetric $l \times l$ matrix with elements $D_{ij} = y_i y_j \mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j$.

Acknowledgment

We thank Professor Bruce Webster of Poultry Science at the University of Georgia (UGA) for helps with the test of live birds at the UGA, and staff of the GTRI Food Processing Division for helps in vision hardware. Birds used in these experiments were provided by Gold Kist Inc.

References

[1] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," *Proc. IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), Maui, HI, USA, 1991.

[2] A. Pentland, B. Moghaddam, and T. Starner, "View-based and modular eigenspaces for face recognition," *CVPR'94,* Seattle, USA, 1994.

[3] X. Mu, M. Artiklar, M. H. Hassoun, and P. Watta, "An RCE-based associative memory with application to human face recognition," Proc. of Int. Joint Conf. on Neural Networks, Portland, OR, U. S, 2003.

[4] H. A. Rowley, S. Baluja, and T. Kanade, "Rotation invariant neural network-based face detection," *CVPR'98*, Santa Barbara, USA, 1998.

[5] D. Chai and K. N. Ngan, "Face segmentation using skin-color map in videophone applications," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 9, pp. 551, 1999.

[6] N. Habili, L. C. Chew, and A. Moini, "Segmentation of the face and hands in sign language video sequences using color and motion cues," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 14, pp. 1086, 2004.

[7] F. Hsin-Chia, P. S. Lai, R. S. Lou, and H. T. Pao, "Face detection and eye localization by neural network based color segmentation," *Proc. of IEEE Signal Processing Society Workshop*, Sydney, Australia, 2000.

[8] R. Chellappa, C. L. Wilson, and S. Sirohey, "Human and machine recognition of faces: a survey," *Proc. of the IEEE*, vol. 83, 1995.

[9] K._M. Lee, W. Daley and Q. Li, Artificial Color Contrast for Machine Vision and its Effects on Feature Detection, Submitted to AIM2005.

[10] D. K Reilly, L. N. Cooper, and C. Elbaum, "A Neural Model for Category Learning." Biological Cybernetics, Vol. 45, pp35-41, 1982.

[11] T. Fang, W. Fang, P. K. Willett, K. R. Pattipati, and E. H. Jordan, "Signal processing and neural network toolbox and its application to failure diagnosis and prognosis," *Proc. SPIE*, vol. 4389, pp. 121, 2001.

[12] S. Keyvan, "Traditional signal pattern recognition versus artificial neural networks for nuclear plant diagnostics," *Progress in Nuclear Energy*, vol. 39, pp. 1, 2001.

[13] T. Olmez and Z. Dokur, "Application of InP neural network to ECG beat classification," *Neural Computing & Applications*, vol. 11, pp. 144, 2003.

[14] S.-M. Roan, C.-C. Chiang, and H.-C. Fu, "Fuzzy RCE neural network," San Francisco, CA, Publ by IEEE, Piscataway, NJ, USA, 1993.

[15] M. L. Yuan and M. Xie, "An incremental representation of conceptual symbols using RCE neural network," Proc. of the 2nd Int. Conf. on Development and Learning, Cambridge, MA, USA, 2002.

[16] Y. Xiaoming, L. Jiangzhou, and X. Ming, "Vision-based human-vehicle interaction and skill training," Proceedings of 10th Int. Conf. on Advanced Robotics. Budapest, Hungary, 2001.

[17] K.-M Lee, J. Joni and X Yin. "Imaging and Motion Prediction for an Automated Live-Bird Transfer Process," *Proc. of the ASME IMECE DSC* November 5-10, Orlando. 2000.

[18] M. J. Hudak, "RCE networks: an experimental investigation," *IJCNN'91*, Seattle, USA, 1991.

[19] T. Olmez, E. Yazgan, and O. K. Ersoy, "Modified restricted Coulomb energy neural network," *Electronics Letters*, vol. 29, pp. 1963, 1993.

[20] C. Campbell, "Kernel methods: A survey of current techniques," *Neurocomputing*, vol. 48, pp. 63, 2002.

[21] C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining & Knowledge Discovery*, Vol. 2, pp. 121, 1998.

[22] B. Schèolkopf, C. J. C. Burges, and A. J. Smola, *Advances in kernel methods: support vector learning*. Cambridge, Mass.: MIT Press, 1999.

[23] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: An application to face detection," *CVPR'97*, San Juan, USA, 1997.

[24] I.T Jolliffe. *Principal component analysis*. New York : Springer-Verlag, c1986

[25] V. Vapnik and A. Lerner. "Pattern recognition using generalized portrait method." Automation and Remote Control, 24. 1963

[26] M. Aizerman., E. Braverman and L. Rozonoer "Theoretical foundations of the potential method in pattern recognition learning." Journal of the ACM, 44(4):615-631, 1997.

[27] B.E Boser., I.M. Guyon and V.N. Vapnik "A training algorithm for optimal margin classifiers." Proceedings of the 5th Annual ACM workshop on computational learning theory, Pittsburgh, PA, 1992.